862.C1901

PATENT APPLICATION

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:	)	
	:	Examiner: N/Y/A
MIYUKI ENOKIDA, ET AL.	)	
	:	Group Art Unit: 2771
Application No.: 09/558,656	)	
	:	
Filed: April 26, 2000	)	
	:	
For: DATA PROCESSING METHOD AND DATA PROCESSING DEVICE	)	
	:	
	)	Date: September 8, 2000

Commissioner for Patents
Washington, D.C. 20231

CLAIM TO PRIORITY

Sir:

Applicants hereby claim priority under the International Convention and all rights to which they are entitled under 35 U.S.C. § 119 based upon the following Japanese Priority Applications:

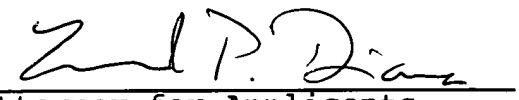
2000-109923 filed on April 11, 2000; and

11-120713 filed on April 27, 1999.

Certified copies of the priority documents are enclosed.

Applicants' undersigned attorney may be reached in our New York office by telephone at (212) 218-2100. All correspondence should continue to be directed to our address given below.

Respectfully submitted,



Attorney for Applicants

Registration No. 29,296

FITZPATRICK, CELLA, HARPER & SCINTO
30 Rockefeller Plaza
New York, New York 10112-3801
Facsimile: (212) 218-2200

NY_MAIN 81684 v2

CFM 1901 US
USAN 09/558,656

日 本 国 特 許 庁
PATENT OFFICE
JAPANESE GOVERNMENT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２０００年 ４月１１日

出願番号
Application Number: 特願２０００－１０９９２３

出願人
Applicant (s): キヤノン株式会社

CERTIFIED COPY OF
PRIORITY DOCUMENT

２０００年 ５月１９日

特許庁長官
Commissioner,
Patent Office 近藤隆彦

出証番号 出証特２０００－３０３６１２９

【書類名】　特許願
【整理番号】　4190007
【提出日】　平成12年 4月11日
【あて先】　特許庁長官殿
【国際特許分類】　G06F　3/00
【発明の名称】　データ処理方法及び装置及び記憶媒体
【請求項の数】　61
【発明者】
【住所又は居所】　東京都大田区下丸子３丁目３０番２号　キヤノン株式会社内
【氏名】　榎田　幸
【発明者】
【住所又は居所】　東京都大田区下丸子３丁目３０番２号　キヤノン株式会社内
【氏名】　草間　澄
【特許出願人】
【識別番号】　000001007
【氏名又は名称】　キヤノン株式会社
【代理人】
【識別番号】　100076428
【弁理士】
【氏名又は名称】　大塚　康徳
【電話番号】　03-5276-3241
【選任した代理人】
【識別番号】　100101306
【弁理士】
【氏名又は名称】　丸山　幸雄
【電話番号】　03-5276-3241

【選任した代理人】

【識別番号】　100115071

【弁理士】

【氏名又は名称】　大塚　康弘

【電話番号】　03-5276-3241

【先の出願に基づく優先権主張】

【出願番号】　平成11年特許願第120713号

【出願日】　平成11年 4月27日

【手数料の表示】

【予納台帳番号】　003458

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　0001010

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　データ処理方法及び装置及び記憶媒体

【特許請求の範囲】

【請求項１】　ディレクトリ構造でファイルを管理するファイル管理システムにおいてディレクトリに対応するディレクトリデータを読み込む第１読込工程と、

前記ディレクトリデータに付与すべきメタデータを読み込む第２読込工程と、

前記第１読込工程で読み込まれたディレクトリデータの後に、前記第２読込工程で読み込まれたメタデータを接続する接続工程と、

前記接続工程によって得られたデータの全体をディレクトリデータファイルとして出力する出力工程と

を備えることを特徴とするデータ処理方法。

【請求項２】　前記第２読込工程で読み込まれたメタデータが、所定のデータ記述言語における適正な形式で記述されているか否かを判定する判定工程を更に備え、

前記接続工程は、前記判定工程で適正な形式で記述されていると判定された場合に、前記メタデータを前記ディレクトリデータの後に接続する

ことを特徴とする請求項１に記載のデータ処理方法。

【請求項３】　前記判定工程は、前記メタデータが前記所定のデータ記述言語としての正当性を満足するか否かに基づいて判定する

ことを特徴とする請求項２に記載のデータ処理方法。

【請求項４】　前記判定工程は、前記メタデータがデータ記述言語が文法的に正しいか否かに基づいて判定する

ことを特徴とする請求項２に記載のデータ処理方法。

【請求項５】　ディレクトリデータファイルのデータにメタデータが登録されているか否かを判別するデータ処理方法であって、

ディレクトリデータファイルを読み込む読込工程と、

前記読込工程で読み込まれたディレクトリデータファイルのデータを末尾より検査し、所定のデータ記述言語における適正な形式で記述されたデータが存在す

るか否かを判定することにより、該データに含まれるメタデータを判別する判別工程と

を備えることを特徴とするデータ処理方法。

【請求項６】　前記判別工程においてメタデータが判別された場合、判別されたメタデータを抽出して出力する出力工程を更に備える

ことを特徴とする請求項５に記載のデータ処理方法。

【請求項７】　前記出力工程は、前記抽出されたメタデータに基づく表示を行う

ことを特徴とする請求項６に記載のデータ処理方法。

【請求項８】　前記出力工程は、前記抽出されたメタデータを、前記所定のデータ記述言語に基づいて所定の処理を実行するためのツールに提供する

ことを特徴とする請求項６に記載のデータ処理方法。

【請求項９】　前記判別工程は、

前記所定のデータ記述言語によって規定された末尾文字列が前記データの末尾に存在するか否かをチェックするチェック工程と、

該末尾文字列が存在する場合に前記所定のデータ記述言語に規定された先頭文字列を該データの先頭へ向かって検索する検索工程とを備え、

前記末尾文字列と前記先頭文字列によって挟まれたデータが存在する場合に、このデータをメタデータと判別する

ことを特徴とする請求項５に記載のデータ処理方法。

【請求項１０】　前記判別工程は、前記末尾文字列と前記先頭文字列によって挟まれたデータが、前記所定のデータ記述言語における適正な形式を有するか否かを検査する検査工程を更に備える

ことを特徴とする請求項９に記載のデータ処理方法。

【請求項１１】　前記検査工程は、前記所定のデータ記述言語としての正当性を満足するかの判断を含めて検査する

ことを特徴とする請求項１０に記載のデータ処理方法。

【請求項１２】　前記検査工程は、前記所定のデータ記述言語が文法的に正しいか否かの判断を含めて検査する

ことを特徴とする請求項１０に記載のデータ処理方法。

【請求項１３】　処理対象データが属するディレクトリに対応するディレクトリデータを読み込む読込工程と、

前記読込工程で読み込まれたディレクトリデータよりメタデータを抽出する抽出工程と、

前記抽出工程で抽出されたメタデータに基づいて、前記処理対象データへメタデータを付加する付加工程と

を備えることを特徴とするデータ処理方法。

【請求項１４】　前記付加工程は、

前記処理対象データにメタデータが登録されているか否かを判定する判定工程を備え、

メタデータが登録されていないと判定された場合に、前記処理対象データに前記抽出工程で抽出されたメタデータを付加する

ことを特徴とする請求項１３に記載のデータ処理方法。

【請求項１５】　前記付加工程は、

前記処理対象データにメタデータが登録されているか否かを判定する判定工程と、

メタデータが登録されていると判定された場合に、前記処理対象データからメタデータを分離する分離工程と、

前記分離工程で得られたメタデータと、前記抽出工程で得られたメタデータとに基づいて新たなメタデータを生成する生成工程とを備え、

前記生成工程によって生成されたメタデータを、前記分離工程によってメタデータが分離された後の前記処理対象データに付加する

ことを特徴とする請求項１３に記載のデータ処理方法。

【請求項１６】　前記生成工程は、前記分離工程で得られたメタデータと、前記抽出工程で得られたメタデータとに含まれる全てのデータ項目が含まれるように新たなメタデータを生成する

ことを特徴とする請求項１５に記載のデータ処理方法。

【請求項１７】　前記処理対象データは、他のディレクトリにコピーされた

データである

ことを特徴とする請求項１３に記載のデータ処理方法。

【請求項１８】　前記処理対象データは、他のディレクトリに移動されたデータである

ことを特徴とする請求項１３に記載のデータ処理方法。

【請求項１９】　前記処理対象データはバイナリデータ部分を含み、

前記付加工程は、前記処理対象データのバイナリデータ部分の後にメタデータを接続する

ことを特徴とする請求項１３に記載のデータ処理方法。

【請求項２０】　前記処理対象データは、画像データ、音声データ、動画像データのいずれかである

ことを特徴とする請求項１３に記載のデータ処理方法。

【請求項２１】　指定されたディレクトリ内に属するデータファイルを読み込む読込工程と、

前記読込工程で読み込まれたデータファイルからメタデータを抽出する抽出工程と、

前記抽出工程で抽出されたメタデータに基づいてディレクトリ用のメタデータを生成する生成工程と、

前記生成工程で生成されたメタデータをディレクトリデータに付加する付加工程と

を備えることを特徴とするデータ処理方法。

【請求項２２】　前記生成工程は、前記抽出工程で抽出したメタデータの全てに含まれる共通のメタデータ項目に基づいて前記ディレクトリに対応するメタデータを生成する

ことを特徴とする請求項２１に記載のデータ処理方法。

【請求項２３】　前記生成工程は、前記抽出工程で抽出したメタデータに最も多く含まれている共通のメタデータ項目に基づいて前記ディレクトリに対応するメタデータを生成する

ことを特徴とする請求項２１に記載のデータ処理方法。

【請求項２４】　新たなディレクトリを生成し、前記生成工程において生成されたディレクトリ用のメタデータに用いられたメタデータ項目を含むメタデータが付加されているデータファイルを該新たなディクレクトリに登録する第１登録工程を更に備え、

前記付加工程は、前記新たなディレクトリに対応するディレクトリデータに前記生成工程で生成されたメタデータを付加する

ことを特徴とする請求項２１に記載のデータ処理方法。

【請求項２５】　新たな別のディクレクトリを生成し、前記生成工程において生成されたディレクトリ用のメタデータに用いられたメタデータ項目が存在しないメタデータが付加されているデータファイルを該新たな別のディクレクトリに登録する第２登録工程を更に備える

ことを特徴とする請求項２１に記載のデータ処理方法。

【請求項２６】　前記データファイルは、画像データファイル、音声データファイル、動画像データファイルのいずれかである

ことを特徴とする請求項２１に記載のデータ処理方法。

【請求項２７】　前記付加工程は、ディレクトリデータの後ろに前記生成工程で生成されたメタデータを接続する

ことを特徴とする請求項２１に記載のデータ処理方法。

【請求項２８】　前記メタデータがＸＭＬのデータ記述言語で記述されている

ことを特徴とする請求項１乃至２７のいずれかに記載のデータ処理方法。

【請求項２９】　前記メタデータがＳＧＭＬのデータ記述言語で記述されている

ことを特徴とする請求項１乃至２７のいずれかに記載のデータ処理方法。

【請求項３０】　前記メタデータがＨＴＭＬのデータ記述言語で記述されている

ことを特徴とする請求項１乃至２７のいずれかに記載のデータ処理方法。

【請求項３１】　ディレクトリ構造でファイルを管理するファイル管理システムにおいてディレクトリに対応するディレクトリデータを読み込む第１読込手

段と、

前記ディレクトリデータに付与すべきメタデータを読み込む第２読込手段と、

前記第１読込手段で読み込まれたディレクトリデータの後に、前記第２読込手段で読み込まれたメタデータを接続する接続手段と、

前記接続手段によって得られたデータの全体をディレクトリデータファイルとして出力する出力手段と

を備えることを特徴とするデータ処理装置。

【請求項３２】　前記第２読込手段で読み込まれたメタデータが、所定のデータ記述言語における適正な形式で記述されているか否かを判定する判定手段を更に備え、

前記接続手段は、前記判定手段で適正な形式で記述されていると判定された場合に、前記メタデータを前記ディレクトリデータの後に接続する

ことを特徴とする請求項３１に記載のデータ処理装置。

【請求項３３】　前記判定手段は、前記メタデータが前記所定のデータ記述言語としての正当性を満足するか否かに基づいて判定する

ことを特徴とする請求項３２に記載のデータ処理装置。

【請求項３４】　前記判定手段は、前記メタデータがデータ記述言語が文法的に正しいか否かに基づいて判定する

ことを特徴とする請求項３２に記載のデータ処理装置。

【請求項３５】　ディレクトリデータファイルのデータにメタデータが登録されているか否かを判別するデータ処理装置であって、

ディレクトリデータファイルを読み込む読込手段と、

前記読込手段で読み込まれたディレクトリデータファイルのデータを末尾より検査し、所定のデータ記述言語における適正な形式で記述されたデータが存在するか否かを判定することにより、該データに含まれるメタデータを判別する判別手段と

を備えることを特徴とするデータ処理装置。

【請求項３６】　前記判別手段においてメタデータが判別された場合、判別されたメタデータを抽出して出力する出力手段を更に備える

ことを特徴とする請求項３５に記載のデータ処理装置。

【請求項３７】　前記出力手段は、前記抽出されたメタデータに基づく表示を行う

ことを特徴とする請求項３６に記載のデータ処理装置。

【請求項３８】　前記出力手段は、前記抽出されたメタデータを、前記所定のデータ記述言語に基づいて所定の処理を実行するためのツールに提供する

ことを特徴とする請求項３６に記載のデータ処理装置。

【請求項３９】　前記判別手段は、

前記所定のデータ記述言語によって規定された末尾文字列が前記データの末尾に存在するか否かをチェックするチェック手段と、

該末尾文字列が存在する場合に前記所定のデータ記述言語に規定された先頭文字列を該データの先頭へ向かって検索する検索手段とを備え、

前記末尾文字列と前記先頭文字列によって挟まれたデータが存在する場合に、このデータをメタデータと判別する

ことを特徴とする請求項３５に記載のデータ処理装置。

【請求項４０】　前記判別手段は、前記末尾文字列と前記先頭文字列によって挟まれたデータが、前記所定のデータ記述言語における適正な形式を有するか否かを検査する検査手段を更に備える

ことを特徴とする請求項３９に記載のデータ処理装置。

【請求項４１】　前記検査手段は、前記所定のデータ記述言語としての正当性を満足するかの判断を含めて検査する

ことを特徴とする請求項４０に記載のデータ処理装置。

【請求項４２】　前記検査手段は、前記所定のデータ記述言語が文法的に正しいか否かの判断を含めて検査する

ことを特徴とする請求項４０に記載のデータ処理装置。

【請求項４３】　処理対象データが属するディレクトリに対応するディレクトリデータを読み込む読込手段と、

前記読込手段で読み込まれたディレクトリデータよりメタデータを抽出する抽出手段と、

前記抽出手段で抽出されたメタデータに基づいて、前記処理対象データへメタデータを付加する付加手段と

を備えることを特徴とするデータ処理装置。

【請求項４４】　前記付加手段は、

前記処理対象データにメタデータが登録されているか否かを判定する判定手段を備え、

メタデータが登録されていないと判定された場合に、前記処理対象データに前記抽出手段で抽出されたメタデータを付加する

ことを特徴とする請求項４３に記載のデータ処理装置。

【請求項４５】　前記付加手段は、

前記処理対象データにメタデータが登録されているか否かを判定する判定手段と、

メタデータが登録されていると判定された場合に、前記処理対象データからメタデータを分離する分離手段と、

前記分離手段で得られたメタデータと、前記抽出手段で得られたメタデータとに基づいて新たなメタデータを生成する生成手段とを備え、

前記生成手段によって生成されたメタデータを、前記分離手段によってメタデータが分離された後の前記処理対象データに付加する

ことを特徴とする請求項４３に記載のデータ処理装置。

【請求項４６】　前記生成手段は、前記分離手段で得られたメタデータと、前記抽出手段で得られたメタデータとに含まれる全てのデータ項目が含まれるように新たなメタデータを生成する

ことを特徴とする請求項４５に記載のデータ処理装置。

【請求項４７】　前記処理対象データは、他のディレクトリにコピーされたデータである

ことを特徴とする請求項４３に記載のデータ処理装置。

【請求項４８】　前記処理対象データは、他のディレクトリに移動されたデータである

ことを特徴とする請求項４３に記載のデータ処理装置。

【請求項４９】　前記処理対象データはバイナリデータ部分を含み、

前記付加手段は、前記処理対象データのバイナリデータ部分の後にメタデータを接続する

ことを特徴とする請求項４３に記載のデータ処理装置。

【請求項５０】　前記処理対象データは、画像データ、音声データ、動画像データのいずれかである

ことを特徴とする請求項４３に記載のデータ処理装置。

【請求項５１】　指定されたディレクトリ内に属するデータファイルを読み込む読込手段と、

前記読込手段で読み込まれたデータファイルからメタデータを抽出する抽出手段と、

前記抽出手段で抽出されたメタデータに基づいてディレクトリ用のメタデータを生成する生成手段と、

前記生成手段で生成されたメタデータをディレクトリデータに付加する付加手段とを

備えることを特徴とするデータ処理装置。

【請求項５２】　前記生成手段は、前記抽出手段で抽出したメタデータの全てに含まれる共通のメタデータ項目に基づいて前記ディレクトリに対応するメタデータを生成する

ことを特徴とする請求項５１に記載のデータ処理装置。

【請求項５３】　前記生成手段は、前記抽出手段で抽出したメタデータに最も多く含まれている共通のメタデータ項目に基づいて前記ディレクトリに対応するメタデータを生成する

ことを特徴とする請求項５１に記載のデータ処理装置。

【請求項５４】　新たなディレクトリを生成し、前記生成手段において生成されたディレクトリ用のメタデータに用いられたメタデータ項目を含むメタデータが付加されているデータファイルを該新たなディクレクトリに登録する第１登録手段を更に備え、

前記付加手段は、前記新たなディレクトリに対応するディレクトリデータに前

記生成手段で生成されたメタデータを付加する

ことを特徴とする請求項５１に記載のデータ処理装置。

【請求項５５】　新たな別のディクレクトリを生成し、前記生成手段において生成されたディレクトリ用のメタデータに用いられたメタデータ項目が存在しないメタデータが付加されているデータファイルを該新たな別のディクレクトリに登録する第２登録手段を更に備える

ことを特徴とする請求項５１に記載のデータ処理装置。

【請求項５６】　前記データファイルは、画像データファイル、音声データファイル、動画像データファイルのいずれかである

ことを特徴とする請求項５１に記載のデータ処理装置。

【請求項５７】　前記付加手段は、ディレクトリデータの後ろに前記生成手段で生成されたメタデータを接続する

ことを特徴とする請求項５１に記載のデータ処理装置。

【請求項５８】　前記メタデータがＸＭＬのデータ記述言語で記述されている

ことを特徴とする請求項３２乃至５７のいずれかに記載のデータ処理装置。

【請求項５９】　前記メタデータがＳＧＭＬのデータ記述言語で記述されている

ことを特徴とする請求項３２乃至５７のいずれかに記載のデータ処理装置。

【請求項６０】　前記メタデータがＨＴＭＬのデータ記述言語で記述されている

ことを特徴とする請求項３２乃至５７のいずれかに記載のデータ処理装置。

【請求項６１】　請求項１乃至３０のいずれかに記載の方法をコンピュータによって実現するための制御プログラムを格納したことを特徴とする記憶媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、データにメタデータを登録、判別することを可能とするデータ処理方法および装置に関する。

【０００２】

【従来の技術】

メタデータ（meta-data）とは、「データに関するデータ」であり、画像データや音声データ等のバイナリデータを説明するデータとして用いられている。しかし、バイナリデータとこれに対応するメタデータが別々のファイルで存在した場合、ファイルの移動やコピーの際に、ユーザはバイナリデータとメタデータとを同時に管理しなければならず、非常にわずらわしいことになる。

【０００３】

そこで一般に、バイナリデータとメタデータの管理を容易にするために、バイナリデータとメタデータを記述する様々な方法が提案されてきた。この種の従来技術は、新しいバイナリフォーマットを規定する方法と、データベースで管理する方法の２つに分けることができる。

【０００４】

まず、新しいバイナリフォーマットを規定する方法の一例をあげると、画像フォーマットではTiff、Exif、Flashpixなどがある。図１６は、バイナリデータにメタデータを埋め込んだフォーマットの概観を示す図である。バイナリデータとしては、例えば画像データが挙げられる。図１６に示されるように、画像のヘッダ部分にメタデータを記述する枠組みを設け、そこにユーザがメタデータを記述するというのが一般的な方法である。このようにメタデータを記述することにより、データの検索・分類が容易になる。また、バイナリデータ内にメタデータを含むようになるので、１つのファイルで管理でき、ファイルの管理は比較的容易になる。

【０００５】

次に、バイナリデータとメタデータをデータベースで管理する方法を説明する。図１７はバイナリデータとメタデータをデータベースで管理する方法を概念的に示した図である。図１７に示されるような、別々のファイルで存在するバイナリデータとメタデータをデータベース等を用いて管理するという方法も広く行われているものである。この場合は既存のバイナリデータが、既存のアプリケーションでそのまま使えるという利点がある。

【０００６】

また、ファイル管理においてディレクトリ構造が用いられ得ることはよく知られている。従って、例えば大量のバイナリ・データファイルを管理する場合は、通常１つのディレクトリの下に全てのバイナリ・データファイルを置くようなことはせず、あるまとまった属性や特性を持つバイナリ・データファイル群毎にサブディレクトリを作成し、その下に複数のバイナリ・データファイルを置くことが多い。この場合、１つのサブディレクトリの下に置かれている各バイナリ・データファイルのメタデータには共通の項目があることになる。しかしながら、現状では、各バイナリ・データファイルの各々が独立に、メタデータを上記２つの方式のいずれかの方法により管理していた。

【０００７】

【発明が解決しようとする課題】

上述したような、メタデータを記述する新フォーマットを規定する方法とデータベースを用いてメタデータを管理する方法のそれぞれに問題がある。

【０００８】

まず、メタデータを記述する新フォーマットを規定した場合には、既存のバイナリデータを当該新フォーマットに変換し、なおかつその新フォーマット内にメタデータを記述しなければならない。更に、その新フォーマット内のメタデータを用いて検索するためには、当該新フォーマット対応のアプリケーションが必要となる。すなわち、メタデータを記述したり利用したりするために、非常に多くのステップと専用の環境が必要になるという問題がある。また、このような新フォーマットのバイナリデータを処理する（例えば画像データであれば画像の再生）ためには、当該フォーマットに対応したアプリケーションが必要であり、既存のアプリケーションでは対応できなくなる。

【０００９】

そのうえ、メタデータの記述方法も新フォーマットにおいて独自に決められたものであり、新フォーマット内のメタデータを利用するアプリケーションを作成するためには、新規にメタデータの検索ルーチンをつくらなければならないという問題もある。さらに、新しい枠組みのメタデータを記述するにはフォーマット

の規定を変更しなければならないという問題点もあった。

【００１０】

一方、データベースを用いてバイナリデータとメタデータを同時に管理する場合、データベースソフトが無ければメタデータの登録も利用もできないという問題があった。また、登録したメタデータを表示するためにも専用のソフトウェアが必要である。更に、バイナリデータをデータベース外に持っていくと、メタデータは付加されず、メタデータのないバイナリデータとなってしまうという問題点もあった。

【００１１】

更に、ディレクトリ構造でバイナリ・データファイルを管理する場合においても、１つのディレクトリの下に置かれているバイナリ・データファイルのメタデータは、各バイナリ・データファイル毎に存在し、それぞれ独立して管理される。このため、複数のバイナリ・データファイルに存在する共通項目についても各バイナリ・データファイル毎独立して管理することになり、管理するメタデータのデータ量が多くなってしまうという問題がある。また、その共通項目を変更する場合には、全てのバイナリ・データファイルのメタデータに対して逐一変更を行わなければならないという欠点もある。

【００１２】

本発明はメタデータの記述・検索に関する上記の問題点に鑑みてなされたものであり、既存のアプリケーションに影響を与えずに、ディレクトリデータにメタデータを登録可能とすることを目的とする。

また、本発明の他の目的は、メタデータの記述に一般的なデータ記述言語を用いることにより、データ記述言語用の既存のツールを利用することを可能とし、対応アプリケーションの開発を容易にすることにある。

また、本発明の他の目的は、メタデータが記述されたディレクトリデータからメタデータを抽出し、例えば検索、参照、変更等の処理に供することを可能とすることにある。

また、本発明の他の目的は、ディレクトリに対してメタデータを登録することにより、メタデータの管理容易化、データ量の削減を図ることにある。

更に、本発明の他の目的は、ある属性あるいは特性などによりグループ化されたディレクトリにより管理されているデータファイルに対して簡便にメタデータを付属可能とすることにある。

【００１３】

また、本発明の他の目的は、ディレクトリからファイルを移動したりコピーした場合であっても、ディレクトリに登録していたメタデータが失われてしまったり、登録したメタデータが無駄になってしまうことを防止することにある。

また、本発明の他の目的は、既に各バイナリデータにメタデータが登録されている場合に、それらメタデータをもとに適切なディレクトリ用のメタデータを生成、付加することを可能にすることにある。

また、本発明の他の目的は、メタデータが登録されたデータファイルから、適切なメタデータを抽出し、そのメタデータをディレクトリデータ登録することによって、検索を高速化することを目的とする。

更に、本発明の他の目的は、データファイルを、ある属性あるいは特性などによりグループ化したディレクトリに簡便に登録する可能とすることを目的とする。

【００１４】

【課題を解決するための手段】

上記の目的の少なくとも一つを達成するための本発明の一態様によるデータ処理方法は例えば以下の構成を備える。すなわち、

ディレクトリ構造でファイルを管理するファイル管理システムにおいてディレクトリに対応するディレクトリデータを読み込む第１読込工程と、

前記ディレクトリデータに付与すべきメタデータを読み込む第２読込工程と、

前記第１読込工程で読み込まれたディレクトリデータの後に、前記第２読込工程で読み込まれたメタデータを接続する接続工程と、

前記接続工程によって得られたデータの全体をディレクトリデータファイルとして出力する出力工程とを備える。

【００１５】

また、上記の目的の少なくとも一つを達成するための本発明の他の態様による

データ処理方法は例えば以下の構成を備える。すなわち、

ディレクトリデータファイルのデータにメタデータが登録されているか否かを判別するデータ処理方法であって、

ディレクトリデータファイルを読み込む読込工程と、

前記読込工程で読み込まれたディレクトリデータファイルのデータを末尾より検査し、所定のデータ記述言語における適正な形式で記述されたデータが存在するか否かを判定することにより、該データに含まれるメタデータを判別する判別工程とを備える。

【００１６】

また、上記の目的の少なくとも一つを達成するための本発明の他の態様によるデータ処理方法は例えば以下の構成を備える。すなわち、

処理対象データが属するディレクトリに対応するディレクトリデータを読み込む読込工程と、

前記読込工程で読み込まれたディレクトリデータよりメタデータを抽出する抽出工程と、

前記抽出工程で抽出されたメタデータに基づいて、前記処理対象データへメタデータを付加する付加工程とを備える。

【００１７】

また、上記の目的の少なくとも一つを達成するための本発明の他の態様によるデータ処理方法は例えば以下の構成を備える。すなわち、

指定されたディレクトリ内に属するデータファイルを読み込む読込工程と、

前記読込工程で読み込まれたデータファイルからメタデータを抽出する抽出工程と、

前記抽出工程で抽出されたメタデータに基づいてディレクトリ用のメタデータを生成する生成工程と、

前記生成工程で生成されたメタデータをディレクトリデータに付加する付加工程とを備える。

【００１８】

また、本発明の他の態様によれば、上記のデータ処理方法を実現するデータ処

理装置が提供される。さらに本発明の他の態様によれば、上記データ処理方法をコンピュータに実現させるためのコンピュータプログラムを格納した記憶媒体が提供される。

【００１９】

【発明の実施の形態】

以下、添付の図面を参照して本発明の好適な実施形態を説明する。

【００２０】

＜第１の実施形態＞

図１は第１の実施形態によるデータ処理装置の構成を示すブロック図である。図１において、１００は読込部であり、スキャナ装置などを用いて画像を読み込む。１０１は入力部であり、ユーザからの指示やデータを入力するもので、キーボードやポインティング装置を含む。１０２は蓄積部であり、バイナリデータやメタデータをディレクトリ構造で蓄積する。蓄積部１０２としては、ハードディスクを用いるのが一般的である。１０３は表示部であり、蓄積部１０２に蓄積されたバイナリデータを表示したり、読込部１００で読み込まれた画像データを表示する。表示部１０３としては、ＣＲＴや液晶表示装置が一般的である。

【００２１】

１０４はＣＰＵであり、上述した各構成の処理のすべてに関わり、ＲＯＭ１０５とＲＡＭ１０６はその処理に必要なプログラム、データ、或いは作業領域をＣＰＵ１０４に提供する。なお、図２のフローチャートを参照して後述する本実施形態の処理手順を実現するための制御プログラムもＲＯＭ１０５に格納されているものとする。もちろん、蓄積部１０２にその制御プログラムを格納しておき、ＣＰＵ１０４による実行に応じてその制御プログラムがＲＡＭ１０６上へロードされるような構成であってもよい。

【００２２】

なお、第１の実施形態のデータ処理装置には上記以外にも、種々の構成要素が設けられているが、本発明の主眼ではないので、その説明については省略する。

【００２３】

つぎに、以上のように構成されたデータ処理装置において、メタデータをディ

レクトリデータに登録する処理について説明する。図２は、第１の実施形態によるメタデータの登録処理を説明するフローチャートである。

【００２４】

図２において、まず、ステップＳ３０１で、ユーザによって指定されたディレクトリのディレクトリデータをメモリ（ＲＡＭ１０６）上に読み込む。これは例えば所望のディレクトリ名をキーボードから入力したり、ポインティング装置（例えばマウス）によって当該ディレクトリを指示することによりなされる。次にステップＳ３０２において、ユーザによって指定された、メタデータが記述されているＸＭＬファイルをメモリ（ＲＡＭ１０６）上に読み込む。このＸＭＬファイルの指定も、キーボードからファイル名を入力したり、ポインティング装置（例えばマウス）で対応するアイコンを指示する等によって行われる。

【００２５】

次にステップＳ３０３で、メタデータを記述したＸＭＬファイルが適正形式のＸＭＬデータであるかを調べる。この適性形式の判定では、ＸＭＬファイルの記述フォーマットを満足しているか（例えば、タグの左右の括弧が正しく対をなしているか、タグ付けの形式が正しいかどうか等の文法的な正しさ）がチェックされる。なお、適性形式のＸＭＬデータであるか否かの判定は、正当なＸＭＬデータであるか否かを含めたチェックであってもよい。ここで、正当なＸＭＬデータか否かの判定は、例えば、ＸＭＬデータがＤＴＤ（Document Type Definition）等のスキーマに従って記述されているかどうか等のチェックを行うことでなされる。

【００２６】

ステップＳ３０３において適正形式のＸＭＬデータでないと判定された場合にはステップＳ３０５に進む。ステップＳ３０５では、ＸＭＬデータにエラーがある旨を表示部１０３に表示し、本処理を終了する。

【００２７】

一方、ステップＳ３０３においてＸＭＬファイルが適正形式のＸＭＬデータであると判定された場合には、処理はステップＳ３０４に進む。ステップＳ３０４では、ステップＳ３０１でメモリ上に読み込まれたディレクトリデータの後ろに

当該メタデータを接続することにより、メタデータの登録を行う。その後、ステップＳ３０６において、メタデータを登録したディレクトリデータを出力し、処理を終了する。なお、ステップＳ３０６におけるデータ出力により、図３に示されるデータ構造を有するメタデータ付ディレクトリデータが１つのディレクトリデータファイルとして蓄積部１０２に格納されることになる。

【００２８】

図３は本実施形態によるディレクトリデータへのメタデータの登録状態を説明する図である。図３に示されるように、ディレクトリデータの最後に（本例では、ディレクトリデータの終端を示す識別子＜ＥＯＦ＞の後に）、ＸＭＬデータで記述されたメタデータを接続する。このようにすることにより、他のアプリケーションには影響を与えずに、メタデータをディレクトリデータに登録することができる。すなわち、一般のアプリケーションはディレクトリデータを参照する場合に、ディレクトリデータの先頭から終端識別子までのデータを用いるので、接続されたメタデータは何等影響を及ぼさないのである。

【００２９】

さらに、メタデータはＸＭＬで記述されているため、このＸＭＬデータ部分を抽出しておくことにより、ＸＭＬデータを理解するツールがあれば、メタデータの追加・変更・参照が可能であり、非常に汎用性に優れている。なお、ディレクトリデータからＸＭＬデータ部分を抽出する処理については第２の実施形態で詳しく説明する。

【００３０】

以上説明したように、第一の実施形態によれば、メタデータをＸＭＬで記述し、ディレクトリデータの最後に接続することにより、既存のアプリケーションに影響を及ぼさずに、既存のディレクトリデータにメタデータを登録することができる。

【００３１】

また。ディレクトリデータにメタデータを登録することによって、当該ディレクトリ下のすべてのファイルにメタデータをつけることが不要となり、その内容の変更も簡単になる。

【００３２】

＜第２の実施形態＞

第１の実施形態においてディレクトリデータにメタデータを登録して、ディレクトリデータファイルとする方法を説明した。第２の実施形態では、ディレクトリデータファイルにメタデータが登録されているかどうかを判別し、登録されている場合にはそのメタデータを抽出する処理について説明する。なお、第２の実施形態におけるデータ処理装置の構成は第１の実施形態（図１）と同様であるのでここでは説明を省略する。

【００３３】

以下、指定されたディレクトリのディレクトリデータファイルに第１の実施形態で説明した如きメタデータが登録されているか否かの判定と、登録されたメタデータを抽出する動作について説明する。図４は第２の実施形態による登録されたメタデータの判別及び抽出手順を示すフローチャートである。なお、本実施形態では、抽出されたメタデータを表示部１０３に表示するが、出力の形態はこれに限らない。例えば、抽出したメタデータを編集や検索等の処理に提供するように構成してもよいことは当業者には明らかであろう。

【００３４】

図４によれば、まず、ステップＳ５０１で、ユーザの指示により、メタデータが登録されているかを判別したいディレクトリデータファイル、即ち処理対象データを指定する。ステップＳ５０１における、処理対象データの指定は、キーボードから当該ディレクトリデータのファイル名を入力したり、対応するアイコンをポインティング装置（マウス）で指示することにより行われる。

【００３５】

次にステップＳ５０２において、指定されたディレクトリデータファイルのデータにＸＭＬで記述されたメタデータが登録されているかどうかを判別する。以下、ステップＳ５０２における判別処理の詳細について図５のフローチャートと、図６の概略図にしたがって説明する。図５は第２の実施形態によるメタデータの判別処理の詳細を説明するフローチャートである。また、図６はメタデータとしてＸＭＬデータが登録されたディレクトリデータのデータ構成例を示す図であ

る。

【００３６】

第１の実施形態で説明したように、メタデータとしてのＸＭＬデータが登録されているディレクトリデータファイル（処理対象データ）のデータ構成は図６のようになっている。したがって、メタデータの有無の判別は以下のように行われる。

【００３７】

図５に示されるように、まず、ステップＳ６０１で、上記ステップＳ５０１で指定されたディレクトリデータファイルのデータ全体（処理対象データの全体）をメモリ（ＲＡＭ１０６）上に読み込む。なお、第１の実施形態のステップＳ３０６によって出力されたデータは一つのディレクトリデータファイルとして管理されるので、ファイル管理システムによってこのデータの全体を読出すことが可能である。

【００３８】

次にステップＳ６０２において、ステップＳ６０１で読み込んだデータの最後に“＜/PhotoXML＞”という文字列があるか調べる。存在しなかった場合はステップＳ６０５に進む。

【００３９】

一方、読み込んだ処理対象データの最後に、“＜/PhotoXML＞”という文字列が存在した場合はステップＳ６０３にすすむ。ステップＳ６０３では“＜/PhotoXML＞”という文字列の前に“＜PhotoXML＞”という文字列が存在するかどうかを調べる。

【００４０】

ステップＳ６０３において“＜PhotoXML＞”という文字列の存在が確認された場合は、ステップＳ６０４にすすむ。ステップＳ６０４においては、メタデータが登録されているものと結論づけ、本処理を終了する。一方、ステップＳ６０３において“＜PhotoXML＞”という文字列の存在が確認されなかった場合には、処理はステップＳ６０５に進む。ステップＳ６０５においては、メタデータは登録されていないものと結論づける。すなわち、ステップＳ６０２で、当該バイナリ

データの最後に文字列“＜/PhotoXML＞”が存在しない場合、ステップＳ６０３で文字列“＜PhotoXML＞”が存在しない場合には、処理はステップＳ６０５に進み、当該処理対象データにメタデータは登録されていないものと結論づける。

【００４１】

なお、ステップＳ６０３において、“＜PhotoXML＞”という文字列の存在が確認された後に、さらにそれらの文字列で囲まれたデータが、ＸＭＬの適正形式で記述されているかを確認するようにしてもよい。更に、この場合、当該データがＸＭＬの正当なデータであるか否かの判定を含めて行うようにしてもよい。適性形式か否かの判定、正当なデータか否かの判定は、第１の実施形態（ステップＳ３０３）で説明したとおりである。

【００４２】

次に、図４のフローチャートにもどる。図５のフローチャートで示される処理によってメタデータが登録されていると結論づけられた場合には、処理はステップＳ５０３に進む。ステップＳ５０３では、文字列“＜PhotoXML＞”と“＜/PhotoXML＞”で囲まれた部分のＸＭＬデータに基づいて登録されているメタデータの内容を表示部１０３に表示し、処理を終了する。一方、ステップＳ５０２でメタデータが登録されていないと判定された場合にはそのまま処理を終了する。

【００４３】

以上説明したように、第２の実施形態によれば、メタデータを登録したディレクトリデータと、通常のディレクトリデータとを、ＸＭＬの記述規則によって判別し、メタデータが登録されているディレクトリデータの場合には、そのメタデータを表示することができる。

【００４４】

すなわち、第２の実施形態によれば、メタデータが登録されたディレクトリデータとメタデータが登録されていないディレクトリデータとを判別するとともに、登録されたメタデータを抽出することが可能となる。従って、メタデータとして既存のデータ記述言語を用いれば、メタデータの参照や編集、検索に際して、当該データ記述言語用の既存のツールをそのまま用いることができ、開発に関する手間も省くことができる。

【００４５】

次に図７を用いて具体的にディレクトリ毎に格納されているバイナリ・データファイルのメタデータを格納する方法について説明する。説明を簡単にするため、ルートディレクトリDirectory 1の下に、Directory 2, Directory 3, Directory 4の３つのサブディレクトリがあり、各サブディレクトリの下に図７に示すようにバイナリ・データファイルが置かれているものとする。

【００４６】

このような場合において、各ディレクトリのディレクトリデータにメタデータを登録すれば、Directory 2の下に置かれているBinary Data File 2-1からBinary Data File 2-3には同一のメタデータが、Directory 3の下のBinary Data File 3-1 とBinary Data File 3-2にも同一のメタデータが、同様にDirectory 4の下に置かれているファイルにも同一のメタデータが夫々保存されることになる。

【００４７】

メタデータの簡単な例を示すと、例えばディジタルカメラで撮影した場合などでは、Directory 2は、３月１０日に撮影し、Directory 3のバイナリデータは３月１１日に撮影したといった情報である。あるいは、「風景」や「人物」等の属性でディレクトリを分けることも考えられる。いずれにせよ、サブディレクトリを作成する時のデータファイルの分類の仕方によりメタデータに格納する項目や内容を決めることができる。

【００４８】

上記実施形態の場合、各バイナリ・データファイル毎にメタデータを保存するのではなく、各サブディレクトリのディレクトリデータにメタデータが格納される。従ってバイナリ・データファイルを読み出そうとするアプリケーションは、そのファイルが属しているディレクトリに付随するメタデータを読み出し、これをそのバイナリ・データファイルのメタデータとして処理するように動作する。

【００４９】

例えば、メタデータに格納されているある項目でバイナリ・データファイルを検索する場合は、本実施形態によれば、ディレクトリデータに付随して格納されているメタデータがディレクトリの下に置かれているバイナリ・データファイル

全てに対して有効なため、ディレクトリのみを検索すればよく、高速に検索処理を行うことが可能となる。

【００５０】

なお、ディレクトリに格納されるデータをバイナリ・データとして説明したが、いかなる種類のデータであってもよいことは明らかである。

【００５１】

＜ファイルシステムによるディレクトリデータの更新について＞

ところで、ディレクトリに対してファイルの追加や削除を行った場合、ファイルシステムはこの処理に応じてディレクトリデータを更新する。しかしながら、メタデータが接続された状態のままで更新を行うと、接続されているメタデータが破損してしまう。従って、上記第１の実施形態によってメタデータが付加されたディレクトリデータを扱うために、ファイルシステムに若干の変更が必要となる。以下、ファイル管理システムによるディレクトリデータの更新処理について説明する。

【００５２】

図８は、本実施形態のファイル管理システムによるディレクトリデータの更新処理を説明するフローチャートである。ディレクトリ内のファイルが更新、追加、削除された場合は、図８に示す処理によってディレクトリデータの更新が行われる。

【００５３】

まず、ステップＳ７０１において、当該ディレクトリのディレクトリデータファイルが読み出される。そして、ステップＳ７０２において、当該ディレクトリデータにメタデータが接続されているか否かを判定する。この判定は、上述の図５に示される手順により行うことができる。そして、メタデータが登録されていると判定された場合は、ステップＳ７０３へ進む。

【００５４】

ステップＳ７０３では、存在が検出されたメタデータを分離し、別の記憶エリアに待避させておく。そして、ステップＳ７０４では、メタデータが分離された後のディレクトリデータに対して、従前と同様の方法で編集を施す。そして、デ

ィレクトリデータの編集を終えたならば、ステップＳ７０５において、先のステップＳ７０３で分離し待避させておいたメタデータを、更新後のディレクトリデータの後に再度接続する。一方、ステップＳ７０２においてメタデータが登録されていなければ、ステップＳ７０６へ進み、即座にディレクトリデータを更新する。

【００５５】

以上の処理において、ステップＳ７０４とステップＳ７０６におけるディレクトリデータの更新処理は、従前のファイル管理システムにおいて実行される処理と同一のものでよい。従って、上述のファイル管理システムの改造は大した作業ではなく、僅かな変更によって本実施形態のディレクトリファイルを取り扱い可能となることが当業者には理解されよう。

【００５６】

以上第１及び第２の実施形態で説明したように、メタデータをディレクトリデータに登録するのでディレクトリ単位でメタデータを登録、管理できる。

また、既存のディレクトリデータの最後にメタデータを接続することにより、既存のアプリケーションに影響を与えずに、ディレクトリデータにメタデータを登録することが可能となる。

更に、メタデータが記述されたディレクトリデータからメタデータを抽出し、例えば検索、参照、変更等の処理に供することが可能となる。従って、メタデータの記述に一般的なデータ記述言語を用いることにより、データ記述言語用の既存のツールを利用することが可能となり、対応アプリケーションの開発が容易である。

また、ディレクトリに対してメタデータを登録することにより、メタデータの管理容易化、データ量の削減が図られる。このため、例えば、メタデータ内のある項目でバイナリデータを検索する場合には、ディレクトリのみ検索を行えばよく、高速に検索することができる。また、メタデータ内の項目に格納されている値を変更する場合には、各バイナリデータファイル毎に行う必要が無く、ディレクトリデータに付属しているメタデータを変更するだけで、そのディレクトリで管理されているバイナリデータ全てに対して変更したことになる。このため、メ

タデータの変更を高速に行える。また、ディレクトリで管理されている複数のバイナリデータに対して共通なメタデータを一つのメタデータで管理することができ、データ量を削減することができる。

【００５７】

＜第３の実施形態＞

第１及び第２の実施形態ではディレクトリデータにメタデータを登録することを示した。ディレクトリにメタデータを登録することによって、当該ディレクトリ下のすべてのファイルにメタデータをつけることが不要となり、その内容の変更も簡単になった。しかしながら、このディレクトリからファイルを移動したりコピーした場合、ディレクトリに登録していたメタデータは失われてしまい、登録したメタデータが無駄になってしまう可能性がある。第３の実施形態では、このような不具合を解消するべく、ディレクトリ内のバイナリデータがディレクトリ外に複製・移動された場合に、そのバイナリデータに自動的にメタデータを登録する。なお、第３の実施形態におけるデータ処理装置の構成は第１の実施形態（図１）と同様であるのでここでは説明を省略する。

【００５８】

次に、第３の実施形態によるバイナリデータへのメタデータの登録方法を図９のフローチャートを参照して説明する。なお、図９のフローチャートでは、ディレクトリ内のバイナリデータがディレクトリ外に複製・移動された場合に、そのバイナリデータにメタデータを自動的に付加する。

【００５９】

まず、ステップＳ１３０１において、ユーザの指示でディレクトリ内の１つのバイナリデータが、他のディレクトリへの複製または移動のために指定される。なお、以下の処理は、操作内容が複製であった場合は、複製後のバイナリデータ（複製元のバイナリデータではない）に対して、操作内容が移動であった場合は、移動後のバイナリデータに対して実行される。次にステップＳ１３０２において、指定されたバイナリデータが属するディレクトリに、第１、第２の実施形態で説明したようなディレクトリメタデータが登録されているかを判別する。ディレクトリメタデータが登録されているか否かの判別は図５及び図６を参照して第

２の実施形態で上述したとおりである。

【００６０】

ステップＳ１３０２による判別の結果、ディレクトリにディレクトリメタデータが登録されている場合はステップＳ１３０３にすすみ、登録されていない場合にはステップＳ１３０４にすすむ。ステップＳ１３０４では、ディレクトリメタデータは登録されていないので、指定されたバイナリデータを複製（または移動）して処理を終える。

【００６１】

一方、ディレクトリメタデータが登録されている場合は、ステップＳ１３０３において、ステップＳ１３０１で指定されたバイナリデータにメタデータが登録されているかどうかを判別する。バイナリデータにメタデータが登録されているか否かを判別する処理について、図１０及び図１１を用いて説明する。図１０は、バイナリデータにメタデータが登録されているか否かを判定するためのフローチャートである。図１１は本実施形態によるバイナリデータへのメタデータの登録状態を説明する図である。

【００６２】

メタデータが登録されているバイナリデータの内部は図１１のようになっている。すなわち、上述のディレクトリデータへのメタデータの付加と同様に、バイナリデータの後尾にＸＭＬ形式で記述されたメタデータが接続されている。したがって、メタデータの有無の判別は、図１０において、まずステップＳ１６０１で、指定されたバイナリデータの全部をメモリ上に読み込む。次にステップＳ１６０２において、読み込んだバイナリデータの最後に“＜/PhotoXML＞”という文字列があるか調べる。 もし存在した場合は、ステップＳ１６０３にすすむ。ステップＳ１６０３において、“＜/PhotoXML＞”という文字列の前に“＜PhotoXML＞”という文字列が存在するか調べる。もし存在した場合は、ステップＳ１６０４に進み、当該バイナリデータにメタデータが登録されているものと結論づける。

一方、ステップＳ１６０２で“＜/PhotoXML＞”という文字列が存在しないと判断された場合、或いはステップＳ１６０３で“＜PhotoXML＞”という文字列が

存在しないと判断された場合は、ステップＳ１６０５にすすみ、当該バイナリデータにメタデータは登録されていないと結論づける。

【００６３】

なお、ステップＳ１６０３において、“＜PhotoXML＞”という文字列の存在が確認された後に、さらにそれらの文字列で囲まれたデータが、ＸＭＬの適正形式で記述されているかを確認するようにしてもよい。更に、この場合、当該データがＸＭＬの正当なデータであるか否かの判定を含めて行うようにしてもよい。適性形式か否かの判定、正当なデータか否かの判定は、第１の実施形態（ステップＳ３０３）で説明したとおりである。

【００６４】

以上の判別処理によりバイナリデータにメタデータが登録されていないと判別された場合にはステップＳ１３０８にすすむ。ステップＳ１３０８に処理が進んだ場合、ディレクトリにはディレクトリメタデータが登録され、バイナリデータにはメタデータが登録されていない状態である。従って、ディレクトリメタデータをバイナリデータに登録し、バイナリデータを複製(移動)して処理を終了する。なお、バイナリデータへのメタデータの登録は、第１の実施形態で説明したディレクトリデータへのメタデータの登録と同様の方法で行われる。すなわち、バイナリデータの末尾にメタデータを接続して、図１１に示すような形態のデータを生成する。

【００６５】

一方、ステップＳ１３０３においてバイナリデータにメタデータが登録されていると判別された場合は、ステップＳ１３０５へ進む。ステップＳ１３０５においては、ディレクトリに登録されていたディレクトリメタデータとバイナリデータに登録されていたメタデータとを比較する。次に、ステップＳ１３０６において、ステップＳ１３０５で比較した結果、ディレクトリメタデータとメタデータすべてが含まれるようなメタデータを作成する。

【００６６】

ステップＳ１３０６におけるメタデータの生成処理について詳しく説明する。ステップＳ１３０６では、図１２に示されるように、ディレクトリメタデータ１

８０１とメタデータ１８０２をあわせて、それらの内容すべてが含まれるようなメタデータ１８０３を作成する。

【００６７】

以上のようにして作成したメタデータをバイナリデータに新たに登録し、バイナリデータを複製(移動)して、処理を終了する。

【００６８】

以上説明したように、第３の実施形態によればディレクトリに登録されたディレクトリメタデータを、ディレクトリ内のバイナリデータの複製・移動の際にメタデータとして登録することによって、ディレクトリ内のファイル（例えばバイナリデータ）が別の場所に移動しても、当該バイナリデータに対応するメタデータが失われることはない。

【００６９】

また、あらかじめメタデータがバイナリデータに登録してあった場合でも、ディレクトリメタデータと合成して、ディレクトリとバイナリデータ両方のメタデータを登録することができる。

【００７０】

さらに、メタデータはデータ記述言語で記述されているので、既存のデータ記述言語専用のツールをそのまま用いることができ、開発に関する手間も省くことができる。

【００７１】

＜第４の実施形態＞

第４の実施形態では、あらかじめバイナリデータがメタデータを有しているような場合に、ディレクトリデータに適切なメタデータを登録可能とする。なお、第４の実施形態におけるデータ処理装置の構成は第１の実施形態（図１）と同様であるのでここでは説明を省略する。以下、ディレクトリ内のバイナリデータから適切なディレクトリ用のメタデータを生成する処理について、図１３のフローチャートにしたがって説明する。

【００７２】

図１３において、まず、ステップＳ２３０１において、ユーザの指示で所望の

ディレクトリ（メタデータを生成、付加するべきディレクトリ）が指定される。次にステップＳ２３０２において、ディレクトリ内のデータファイルの一つを選択し、そのデータファイルにメタデータが登録されているかを判別する。データファイルにメタデータが登録されているか否かの判別手順については、図１０および図１１を用いて上述したとおりである。

【００７３】

バイナリデータにメタデータが登録されている場合には、ステップＳ２３０３にすすみ、メタデータが登録されているデータファイル名とメタデータをメモリ上のリスト１に追加する。一方、ステップＳ２３０２において、メタデータが登録されていない場合にはステップＳ２３０４にすすむ。ステップＳ２３０４においては、メタデータが登録されていないデータファイル名をメモリ上のリスト２に追加する。以上の処理を当該ディレクトリ内のすべてのファイルについて実行する(ステップＳ２３０５)。

【００７４】

次に、ステップＳ２３０６において、メタデータとメタデータが登録されているデータファイル名が記録されているリスト１からディレクトリメタデータを生成する。リストからディレクトリメタデータを生成する処理について図１４および図１５を参照して説明する。図１４は、ステップＳ２３０６におけるディレクトリメタデータの生成手順を説明するフローチャートである。また、図１５はステップＳ２３０６におけるディレクトリメタデータの生成を説明する図である。

【００７５】

ステップＳ２４０１において、リスト１に登録されたすべてのメタデータに共通のメタデータ項目があるかを調べる。ここで、すべてのメタデータに共通のメタデータ項目があった場合、ステップＳ２４０２にすすみ、すべてのメタデータに共通のメタデータ項目を抽出し、抽出されたメタデータ項目を用いてディレクトリメタデータを生成して処理を終了する。この処理を図１５を用いて説明する。リスト１（５０１）によれば、img001.jpg、img002.jpg、img003.jpgの３つのイメージファイルにメタデータが付与されているのがわかる（ファイル名は＜Filename＞と＜/Filename＞で囲まれた文字列であり、メタデータは＜PhotoXML＞

と＜/PhotoXML＞で囲まれたデータである）。リスト１（５０１）にリストされている３つのデータファイルのメタデータから共通のメタデータ項目を抽出してディレクトリメタデータを生成すると、ディレクトリメタデータ（５０２）のようになる。

【００７６】

一方、ステップＳ２４０１において全てのメタデータに共通のメタデータ項目が無かった場合には、ステップＳ２４０３において、最も多くのメタデータに登録されているメタデータ項目を用いてディレクトリメタデータを生成する。さらにステップＳ２４０４において、作成されたディレクトリメタデータに含まれるメタデータ項目を含まないメタデータが付加されているデータファイル名を掲載するリスト３を新たに生成する。そして、ステップＳ２４０５において、リスト３に掲載したデータファイル名とそのメタデータをリスト１から削除する。

【００７７】

図１３に戻り、ステップＳ２３０７において、新しいディレクトリを生成し、リスト１に登録されているデータファイル（生成されたディレクトリ用のメタデータに含まれるメタデータ項目と共通するメタデータ項目を少なくとも１つ含むメタデータが登録されているデータファイル）をすべてこの新しいディレクトリに移動する。なお、ステップＳ２３０７において、新しいディレクトリはディレクトリメタデータからの適当なデータを用いて名前をつける。そして、ステップＳ２３０８において、上記ステップＳ２４０４でリスト３が生成されたか否かを判定し、生成されていればステップＳ２３０９へ進み、生成されていなければステップＳ２３０９をスキップする。ステップＳ２３０９では、リスト３にあるデータファイルを“MetaMisc”という名前のディレクトリに移動する。最後に、ステップＳ２３１０において、ステップＳ２３０７で生成した新しいディレクトリにディレクトリメタデータを登録して、処理を終了する。なお、ディレクトリにメタデータを登録する方法は第１の実施形態で説明したとおりであり、図６に示したようなデータ構造となる。

【００７８】

以上のようにしてディレクトリ内に含まれるデータファイルから、ディレクト

リメタデータを自動生成して、ディレクトリにメタデータを登録することができる。

【００７９】

以上説明したように、第４の実施形態によればディレクトリ内のデータファイルからディレクトリメタデータを自動生成することが可能となり、容易にディレクトリにディレクトリメタデータを登録できる。共通のメタデータが無かった場合でも、“MetaMisc”内のバイナリデータから、ディレクトリメタデータを再度自動生成することで、ディレクトリメタデータを次々に生成していくことも可能である。

【００８０】

なお、第４の実施形態において、指定されたディレクトリ内の全てのデータファイルが最後までリスト１に属する場合（これは、当該ディレクトリ内の全てのデータファイルにメタデータが登録されており、それら全てのメタデータが共通のメタデータ項目を有する場合である）には、新たなディレクトリを生成せず、当該指定されたディレクトリのディレクトリデータに生成されたメタデータを登録するようにしてもよい。

【００８１】

以上の第４の実施形態の説明から明らかなように、バイナリデータからディレクトリメタデータを自動生成することが可能となる。また、ディレクトリにメタデータをつけることにより、高速な検索を行うことが可能となる。さらに、メタデータは既存のデータ記述言語で記述されているので、既存のデータ記述言語専用のツールをそのまま用いることができ、開発に関する手間も省くことができる。

【００８２】

なお、上記第１～第４の各実施形態では、メタデータとしてＸＭＬデータを用いたがこれに限られるものではない。例えば、ＳＧＭＬやＨＴＭＬ等のデータ記述言語であってもよい。また、各ディレクトリに保持されるファイルのデータとしては、静止画像データ、動画像データ、音声データ等のバイナリデータや、他のいかなる形態のデータであったもよい。

【００８３】

なお、本発明は、複数の機器（例えばホストコンピュータ，インタフェイス機器，リーダ，プリンタなど）から構成されるシステムに適用しても、一つの機器からなる装置（例えば、複写機，ファクシミリ装置など）に適用してもよい。

【００８４】

また、本発明の目的は、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはＣＰＵやＭＰＵ）が記憶媒体に格納されたプログラムコードを読出し実行することによっても、達成されることは言うまでもない。

【００８５】

この場合、記憶媒体から読出されたプログラムコード自体が前述した実施形態の機能を実現することになり、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。

【００８６】

プログラムコードを供給するための記憶媒体としては、例えば、フロッピディスク，ハードディスク，光ディスク，光磁気ディスク，ＣＤ－ＲＯＭ，ＣＤ－Ｒ，磁気テープ，不揮発性のメモリカード，ＲＯＭなどを用いることができる。

【００８７】

また、コンピュータが読出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているＯＳ（オペレーティングシステム）などが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【００８８】

さらに、記憶媒体から読出されたプログラムコードが、コンピュータに挿入された機能拡張ボードやコンピュータに接続された機能拡張ユニットに備わるメモリに書込まれた後、そのプログラムコードの指示に基づき、その機能拡張ボードや機能拡張ユニットに備わるＣＰＵなどが実際の処理の一部または全部を行い、

その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【００８９】

【発明の効果】

以上説明したように、本発明によれば、メタデータをディレクトリデータに登録するのでディレクトリ単位でメタデータを登録、管理できる。

また、本発明によれば、既存のディレクトリデータの最後にメタデータを接続することにより、既存のアプリケーションに影響を与えずに、ディレクトリデータにメタデータを登録することが可能となる。

更に、本発明によれば、メタデータが記述されたディレクトリデータからメタデータを抽出し、例えば検索、参照、変更等の処理に供することが可能となる。従って、メタデータの記述に一般的なデータ記述言語を用いることにより、データ記述言語用の既存のツールを利用することが可能となり、対応アプリケーションの開発が容易である。

また、本発明によれば、ディレクトリに対してメタデータを登録することにより、メタデータの管理容易化、データ量の削減が図られる。このため、例えば、メタデータ内のある項目でバイナリデータを検索する場合には、ディレクトリのみ検索を行えばよく、高速に検索することができる。また、メタデータ内の項目に格納されている値を変更する場合には、各バイナリデータファイル毎行う必要が無く、ディレクトリデータに付属しているメタデータを変更するだけで、そのディレクトリで管理されているバイナリデータ全てに対して変更したことになり、高速に行える。また、ディレクトリで管理されている複数のバイナリデータに対して共通なメタデータを一つのメタデータで管理することができ、データ量を削減することができる。

【００９０】

また、本発明によれば、ディレクトリからファイルを移動したりコピーした場合であっても、ディレクトリに登録していたメタデータが失われてしまったり、登録したメタデータが無駄になってしまうことを防止できる。

また、本発明によれば、既に各バイナリデータにメタデータが登録されている

場合であっても、ディレクトリに適切なメタデータを付加することが可能になる。

また、本発明によれば、メタデータが登録されたデータファイルから、適切なメタデータを抽出し、そのメタデータをディレクトリデータ登録することによって、検索を高速化することができる。

更に、本発明によれば、データファイルを、ある属性あるいは特性などによりグループ化したディレクトリに簡便に登録することができる。

【図面の簡単な説明】

【図１】

第１の実施形態によるデータ処理装置の構成を示すブロック図である。

【図２】

第１の実施形態によるメタデータの登録処理を説明するフローチャートである。

【図３】

実施形態によるディレクトリデータへのメタデータの登録状態を説明する図である。

【図４】

第２の実施形態による登録されたメタデータの判別及び抽出手順を示すフローチャートである。

【図５】

第２の実施形態によるメタデータの判別処理の詳細を説明するフローチャートである。

【図６】

メタデータとしてＸＭＬデータが登録されたディレクトリデータのデータ構成例を示す図である。

【図７】

第２の実施形態によるディレクトリ構造の例を説明する図である。

【図８】

第２の実施形態のファイル管理システムによるディレクトリデータの更新処理

を説明するフローチャートである。

【図９】

第３の実施形態におけるバイナリデータへのメタデータ登録方法を示すフローチャートである。

【図１０】

第３の実施形態におけるバイナリデータにメタデータが登録されているかを判断する方法を示すフローチャートである。

【図１１】

第３の実施形態における、メタデータの登録されているバイナリデータのデータ構成例を示す図である。

【図１２】

第３の実施形態におけるメタデータとディレクトリメタデータを比較し合成する方法の例を説明する図である。

【図１３】

第４の実施形態における、ディレクトリデータに自動生成したディレクトリメタデータを登録する方法を示すフローチャートである。

【図１４】

第４の実施形態における、ディレクトリメタデータを自動生成する方法を示すフローチャートである。

【図１５】

第４の実施形態における、メタデータのリストからディレクトリメタデータを自動生成する例を示す図である。

【図１６】

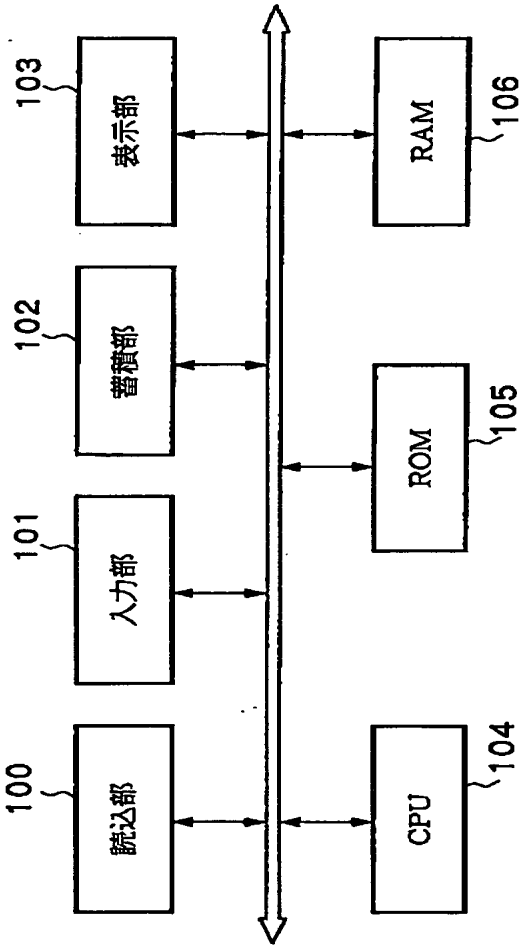
バイナリデータにメタデータを埋め込んだフォーマットの概要を示す図である。

【図１７】

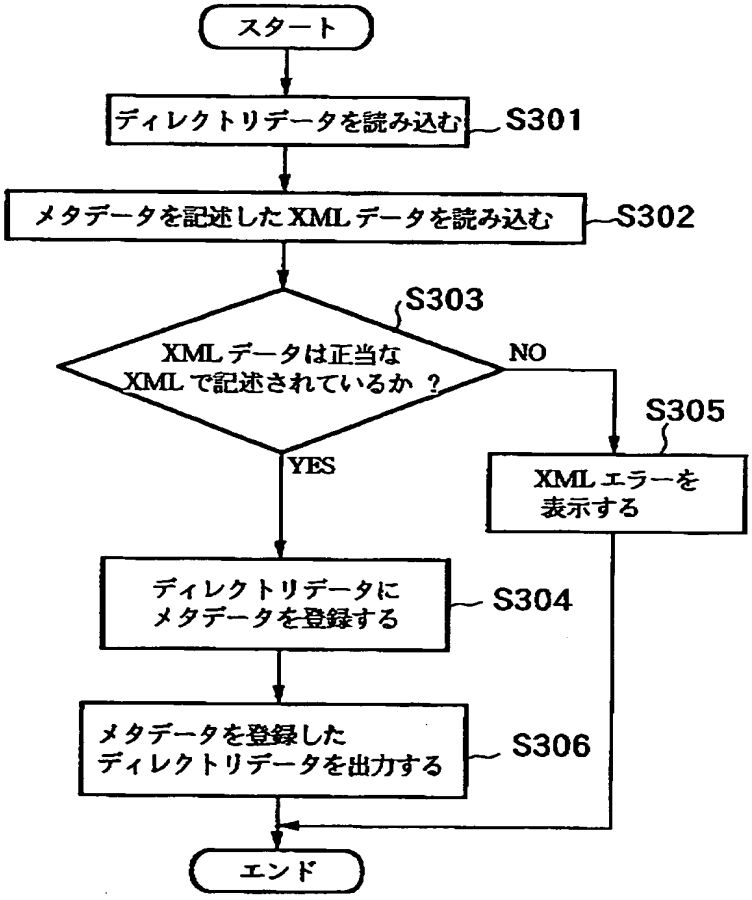
バイナリデータとメタデータをデータベースで管理する方法を概念的に示した図である。

【書類名】　図面

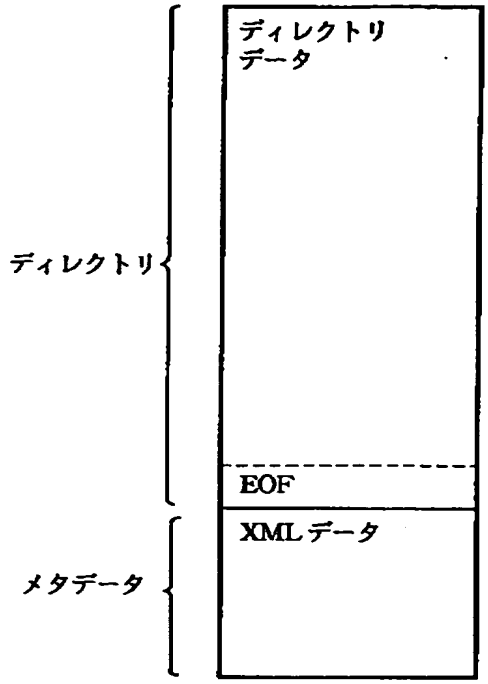
【図１】



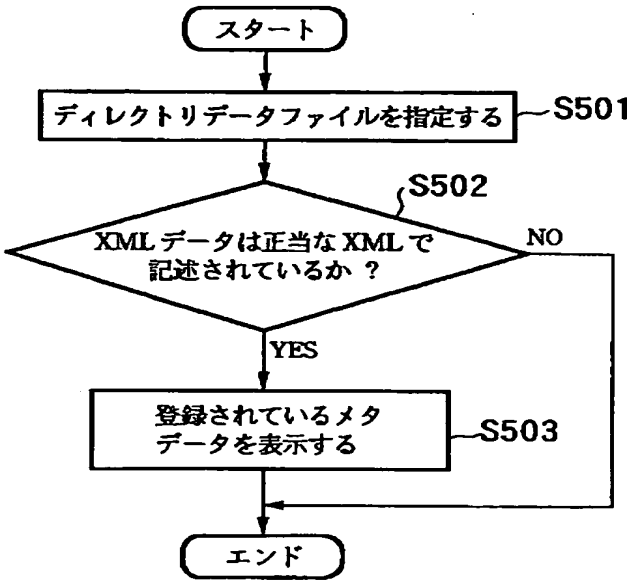
【図２】


スタート
ディレクトリデータを読み込む
S301
メタデータを記述したXMLデータを読み込む
S302
S303
XMLデータは正当な
XMLで記述されているか ？
NO
YES
S305
XMLエラーを
表示する
ディレクトリデータに
メタデータを登録する
S304
メタデータを登録した
ディレクトリデータを出力する
S306
エンド

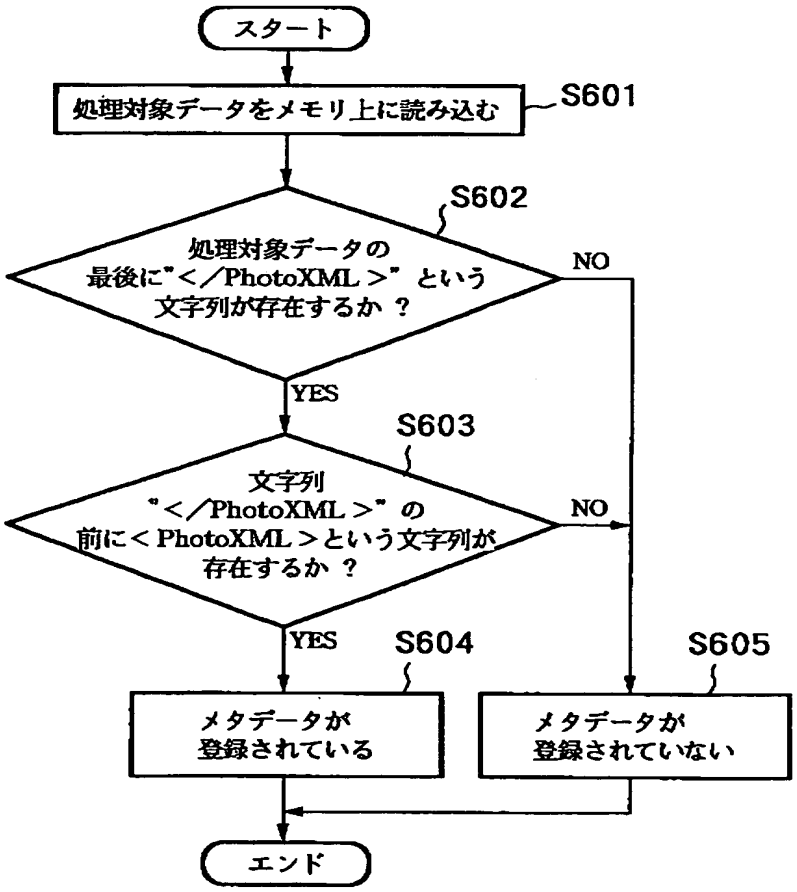
【図３】


ディレクトリ
データ
ディレクトリ
EOF
XMLデータ
メタデータ

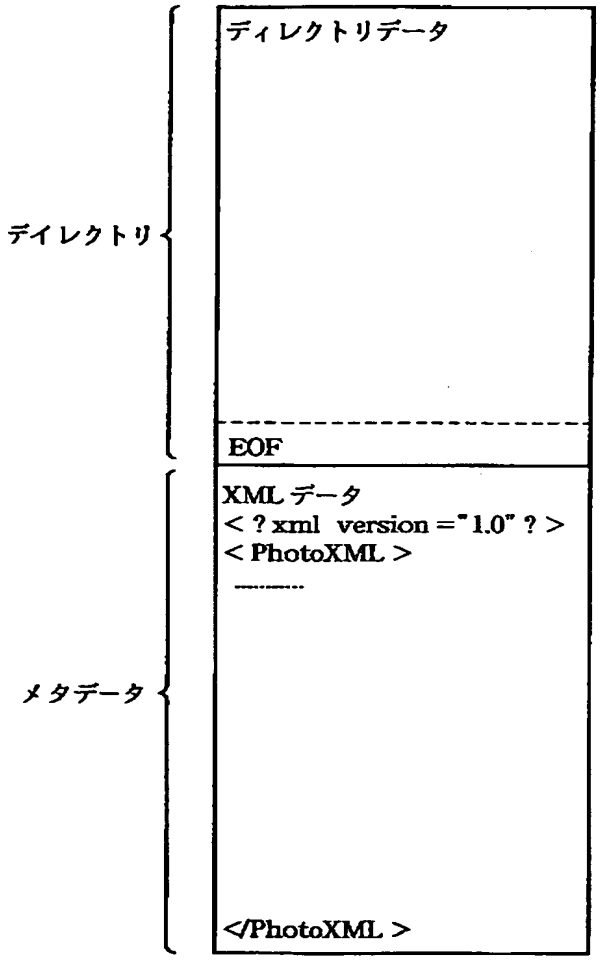
【図４】


スタート
ディレクトリデータファイルを指定する
S501
S502
XMLデータは正当なXMLで
記述されているか ？
NO
YES
登録されているメタ
データを表示する
S503
エンド

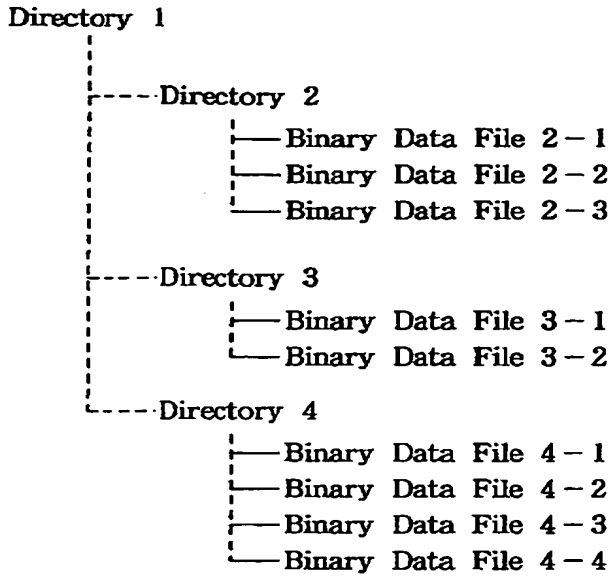
【図５】


スタート
処理対象データをメモリ上に読み込む
S601
S602
処理対象データの
最後に"＜／PhotoXML＞"という
文字列が存在するか ？
NO
YES
S603
文字列
"＜／PhotoXML＞"の
前に＜PhotoXML＞という文字列が
存在するか ？
NO
YES
S604
S605
メタデータが
登録されている
メタデータが
登録されていない
エンド

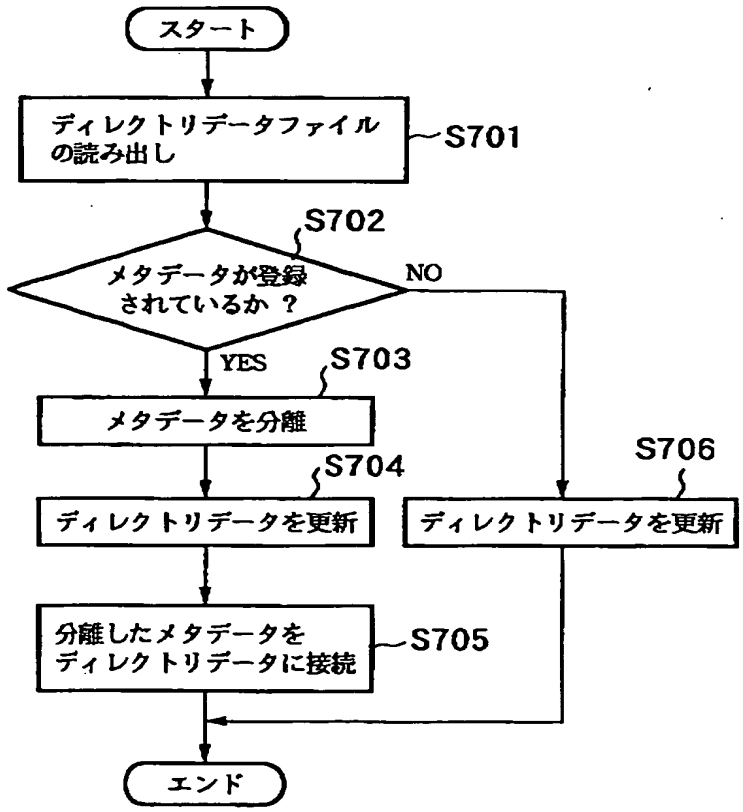
【図６】


ディレクトリデータ
ディレクトリ
EOF
XML データ
< ? xml version ="1.0" ? >
< PhotoXML >
メタデータ
</PhotoXML >

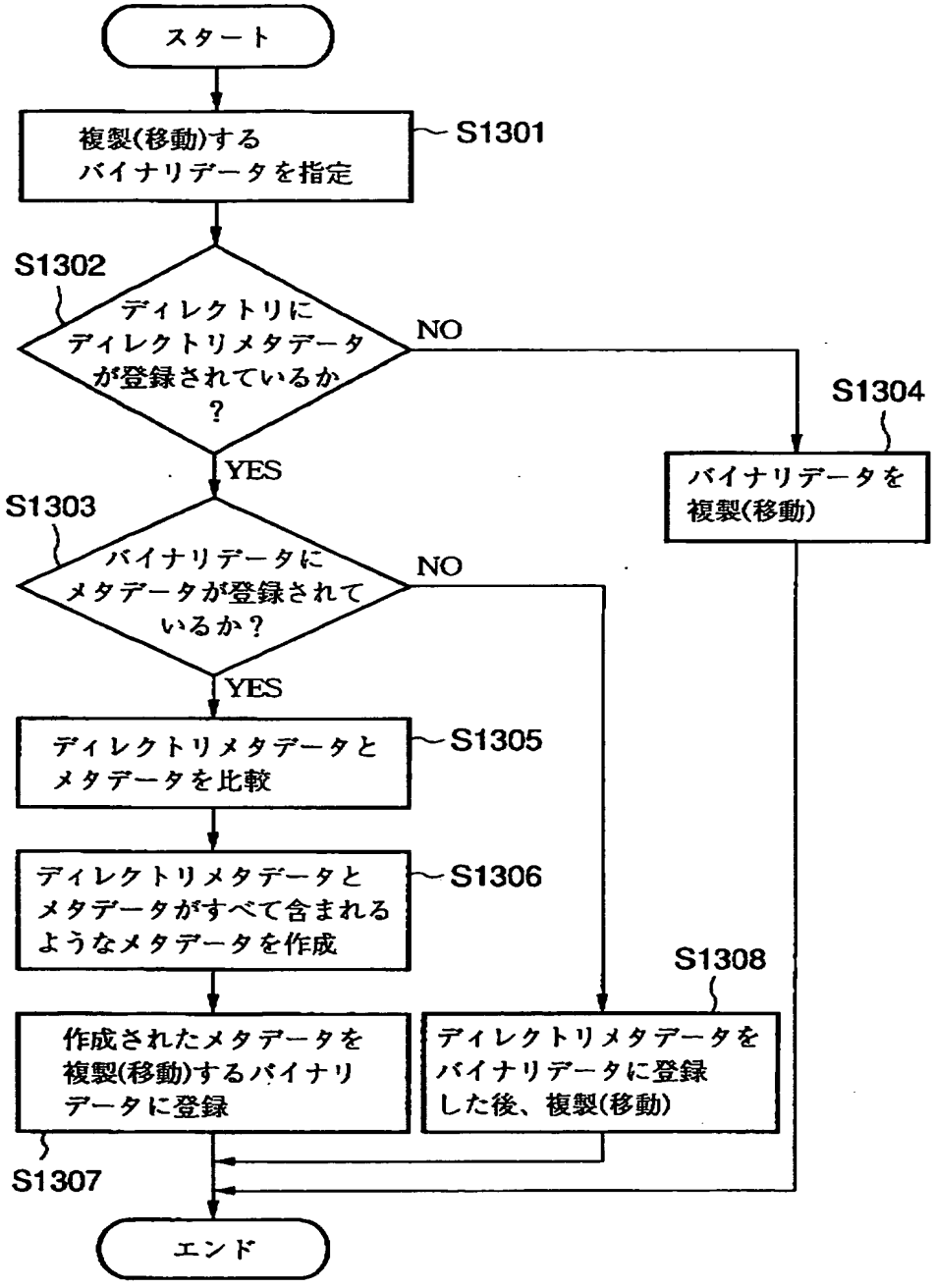
【図７】


Directory 1
Directory 2
Binary Data File 2－1
Binary Data File 2－2
Binary Data File 2－3
Directory 3
Binary Data File 3－1
Binary Data File 3－2
Directory 4
Binary Data File 4－1
Binary Data File 4－2
Binary Data File 4－3
Binary Data File 4－4

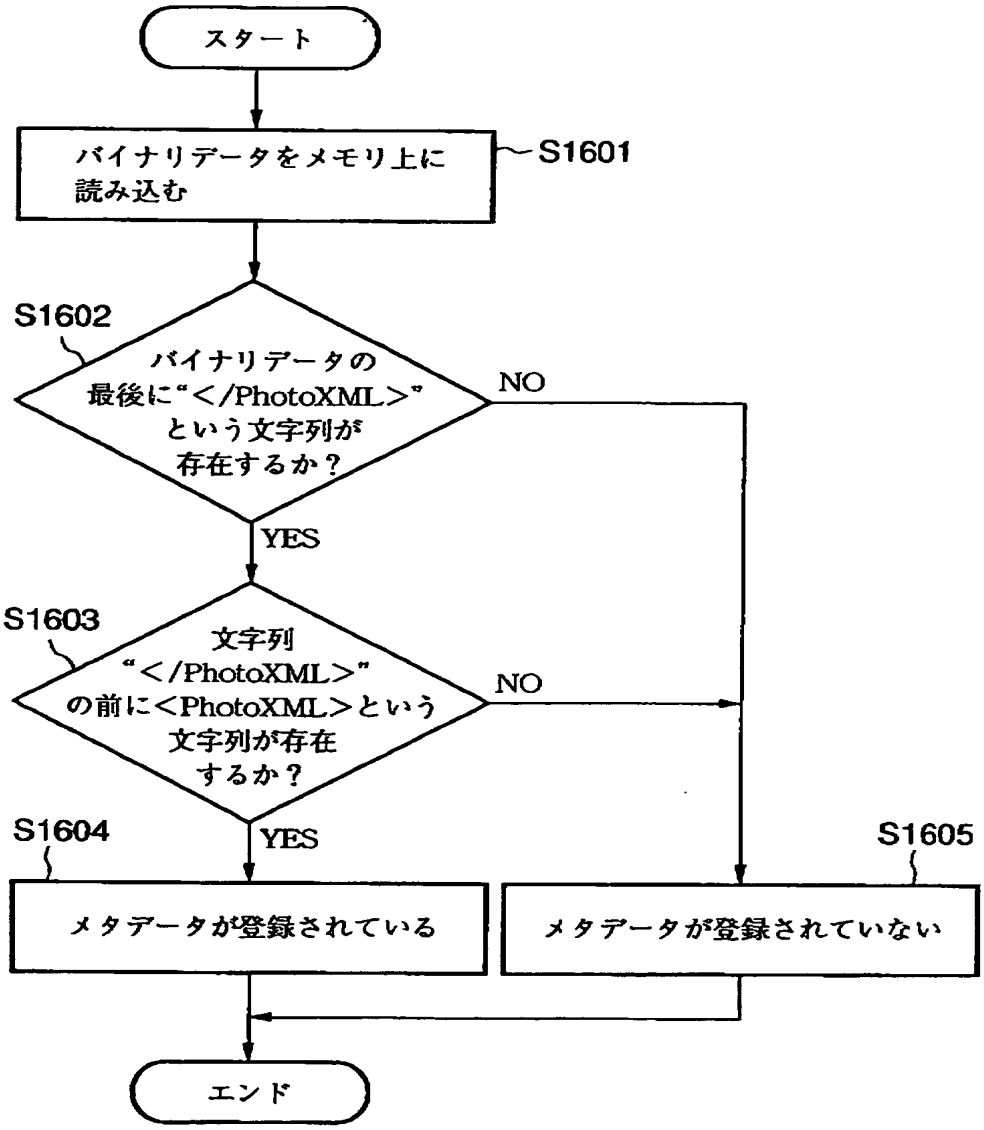
【図８】


スタート
ディレクトリデータファイルの読み出し
S701
S702
メタデータが登録されているか ？
NO
YES
S703
メタデータを分離
S704
ディレクトリデータを更新
S706
ディレクトリデータを更新
分離したメタデータをディレクトリデータに接続
S705
エンド

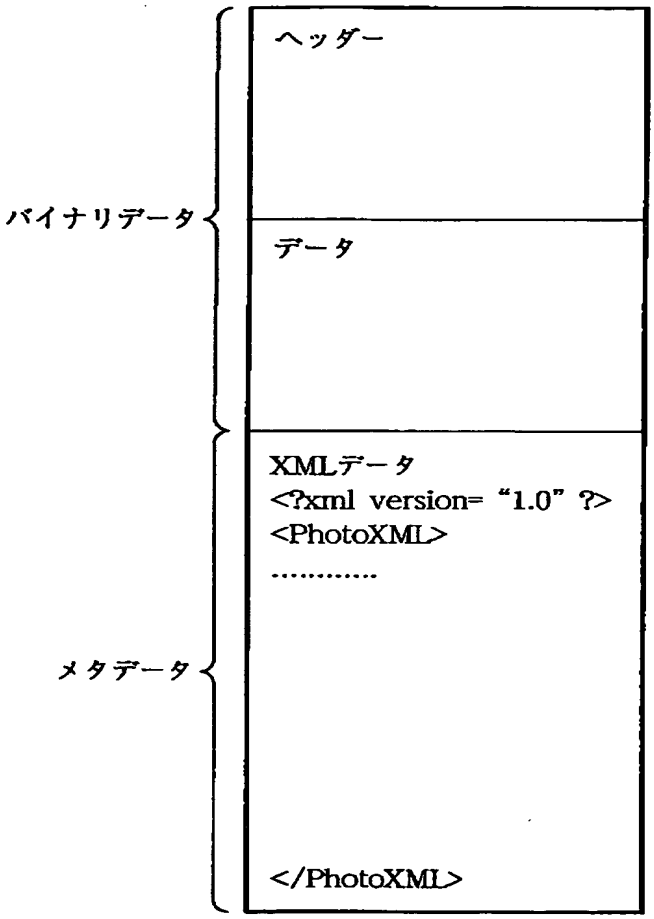
【図９】


スタート
複製(移動)する
バイナリデータを指定
S1301
S1302
ディレクトリに
ディレクトリメタデータ
が登録されているか
？
NO
S1304
バイナリデータを
複製(移動)
YES
S1303
バイナリデータに
メタデータが登録されて
いるか？
NO
YES
ディレクトリメタデータと
メタデータを比較
S1305
ディレクトリメタデータと
メタデータがすべて含まれる
ようなメタデータを作成
S1306
S1308
作成されたメタデータを
複製(移動)するバイナリ
データに登録
ディレクトリメタデータを
バイナリデータに登録
した後、複製(移動)
S1307
エンド

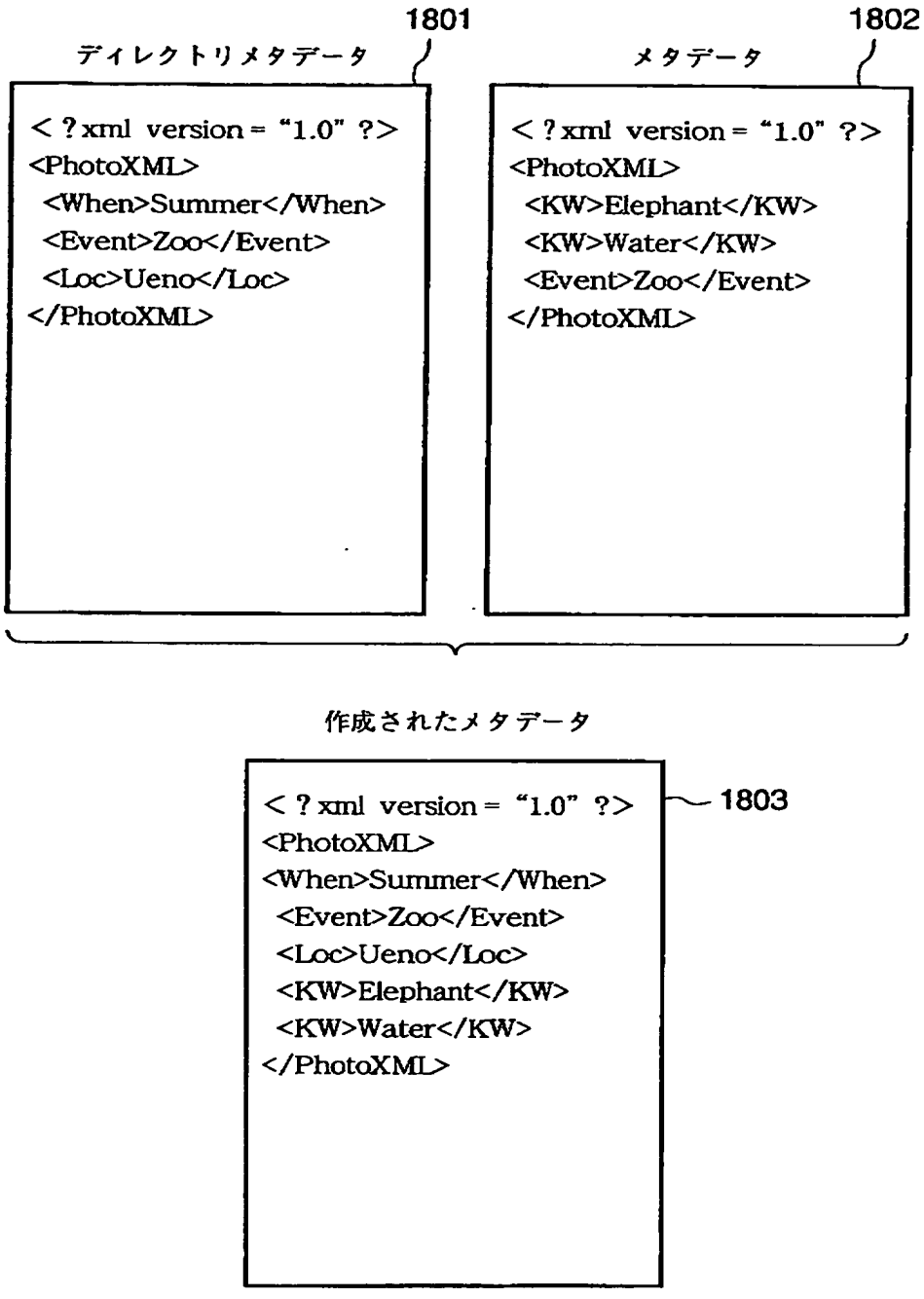
【図１０】


スタート
バイナリデータをメモリ上に読み込む
S1601
S1602
バイナリデータの最後に“＜/PhotoXML＞”という文字列が存在するか？
NO
YES
S1603
文字列“＜/PhotoXML＞”の前に＜PhotoXML＞という文字列が存在するか？
NO
YES
S1604
S1605
メタデータが登録されている
メタデータが登録されていない
エンド

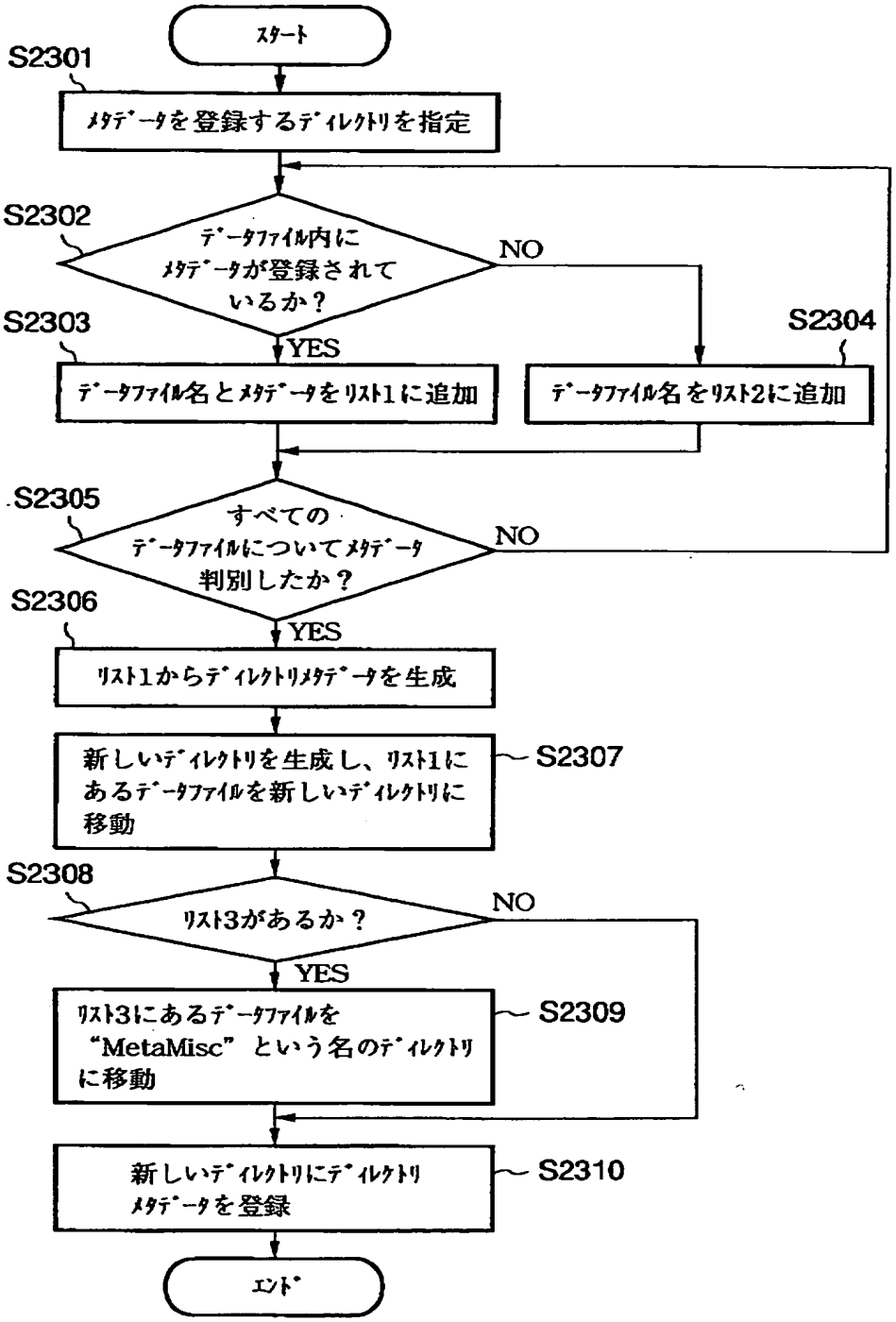
【図１１】


ヘッダー
データ
XMLデータ
<?xml version= “1.0” ?>
<PhotoXML>
.....
</PhotoXML>
バイナリデータ
メタデータ

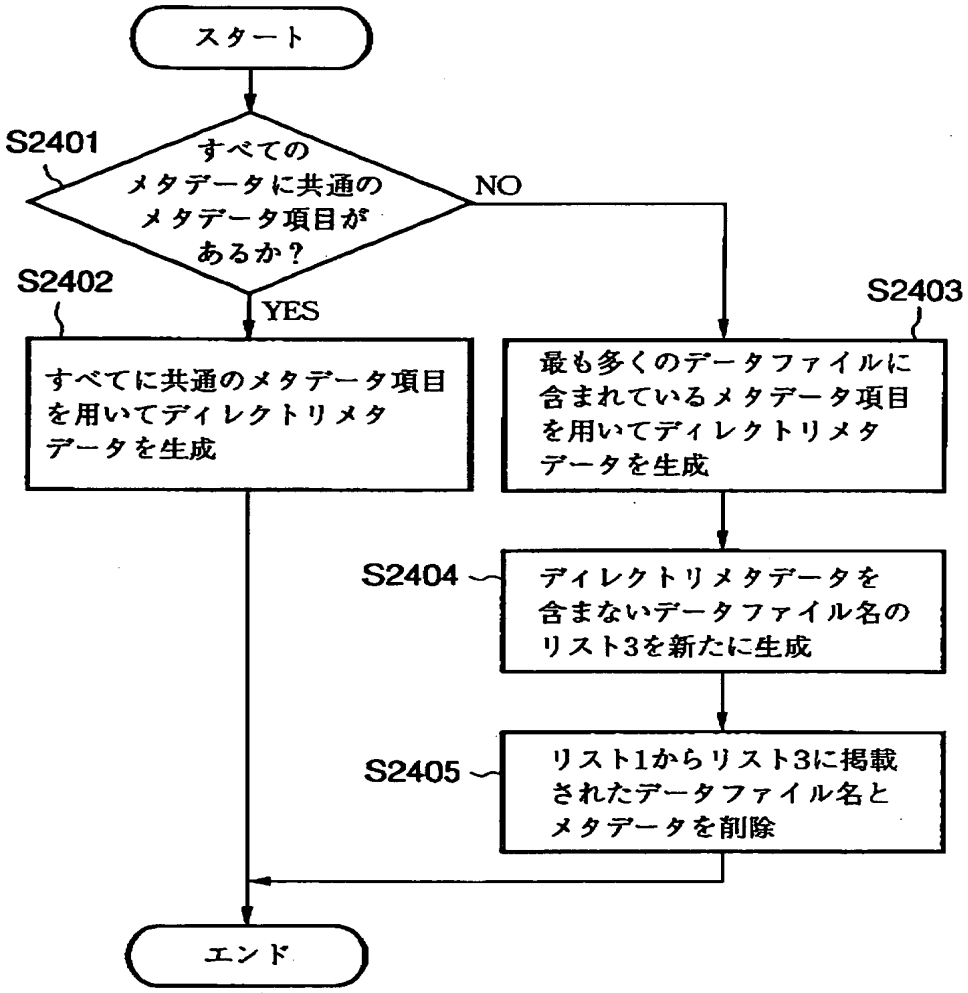
【図１２】


1801
ディレクトリメタデータ
<?xml version = "1.0" ?>
<PhotoXML>
<When>Summer</When>
<Event>Zoo</Event>
<Loc>Ueno</Loc>
</PhotoXML>
1802
メタデータ
<?xml version = "1.0" ?>
<PhotoXML>
<KW>Elephant</KW>
<KW>Water</KW>
<Event>Zoo</Event>
</PhotoXML>
作成されたメタデータ
1803
<?xml version = "1.0" ?>
<PhotoXML>
<When>Summer</When>
<Event>Zoo</Event>
<Loc>Ueno</Loc>
<KW>Elephant</KW>
<KW>Water</KW>
</PhotoXML>

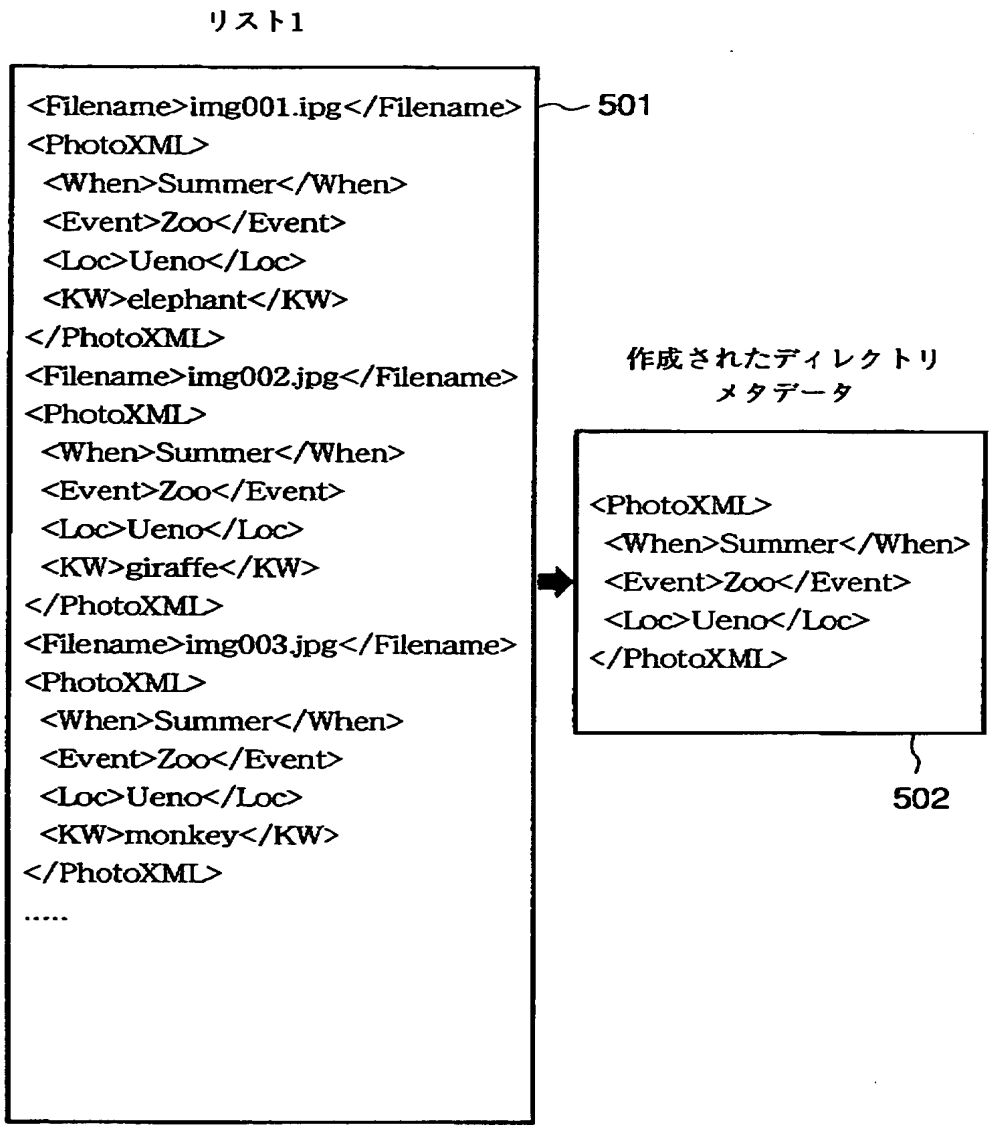
【図１３】


スタート
S2301
メタデータを登録するディレクトリを指定
S2302
データファイル内にメタデータが登録されているか？
NO
YES
S2303
データファイル名とメタデータをリスト1に追加
S2304
データファイル名をリスト2に追加
S2305
すべてのデータファイルについてメタデータ判別したか？
NO
YES
S2306
リスト1からディレクトリメタデータを生成
新しいディレクトリを生成し、リスト1にあるデータファイルを新しいディレクトリに移動
S2307
S2308
リスト3があるか？
NO
YES
リスト3にあるデータファイルを“MetaMisc”という名のディレクトリに移動
S2309
新しいディレクトリにディレクトリメタデータを登録
S2310
エンド

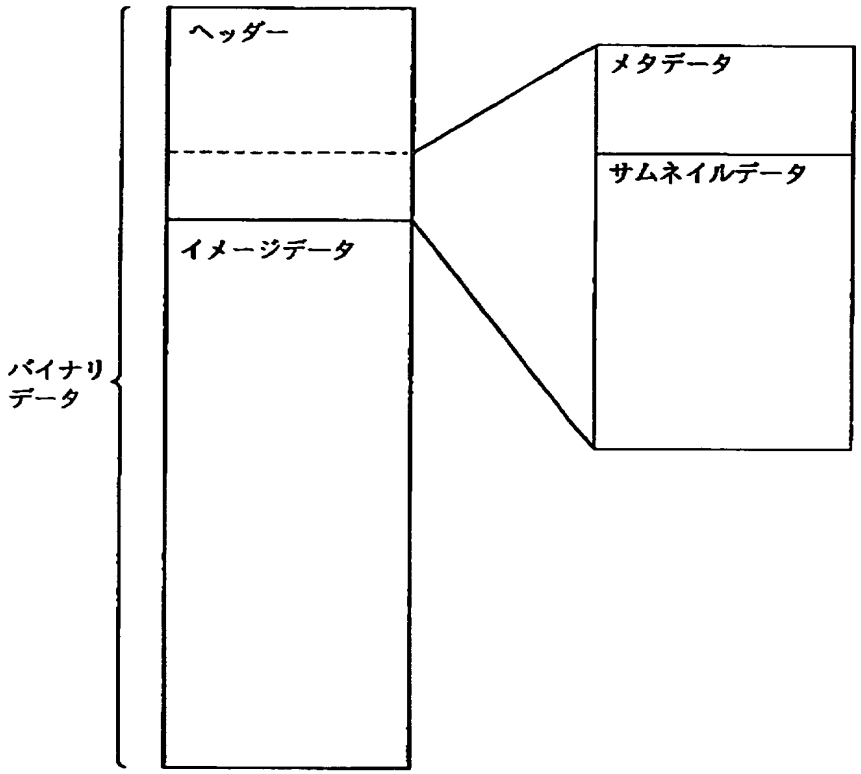
【図１４】


スタート
S2401
すべての
メタデータに共通の
メタデータ項目が
あるか？
NO
YES
S2402
すべてに共通のメタデータ項目
を用いてディレクトリメタ
データを生成
S2403
最も多くのデータファイルに
含まれているメタデータ項目
を用いてディレクトリメタ
データを生成
S2404
ディレクトリメタデータを
含まないデータファイル名の
リスト3を新たに生成
S2405
リスト1からリスト3に掲載
されたデータファイル名と
メタデータを削除
エンド

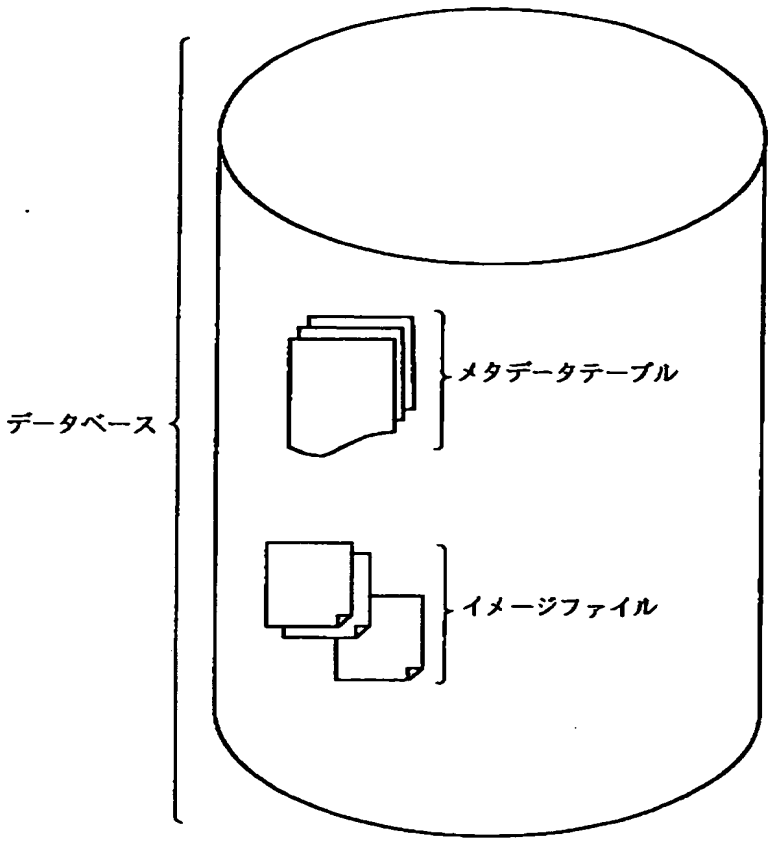
【図１５】


リスト1
<Filename>img001.jpg</Filename>
501
<PhotoXML>
<When>Summer</When>
<Event>Zoo</Event>
<Loc>Ueno</Loc>
<KW>elephant</KW>
</PhotoXML>
<Filename>img002.jpg</Filename>
<PhotoXML>
<When>Summer</When>
<Event>Zoo</Event>
<Loc>Ueno</Loc>
<KW>giraffe</KW>
</PhotoXML>
<Filename>img003.jpg</Filename>
<PhotoXML>
<When>Summer</When>
<Event>Zoo</Event>
<Loc>Ueno</Loc>
<KW>monkey</KW>
</PhotoXML>
.....
作成されたディレクトリ
メタデータ
<PhotoXML>
<When>Summer</When>
<Event>Zoo</Event>
<Loc>Ueno</Loc>
</PhotoXML>
502

【図１６】



【図１７】



【書類名】　要約書

【要約】

【課題】　メタデータをディレクトリデータの最後に接続することにより、ディレクトリで管理されている既存のデータに対し、既存のアプリケーションに影響を与えずに、ディレクトリ単位でメタデータを登録可能とする。

【解決手段】ステップＳ３０１において、ディレクトリ構造でファイルを管理するファイル管理システムによって各ディレクトリ毎に用意されているディレクトリデータを読み込む。ステップＳ３０２で、上記ディレクトリデータに付与すべきメタデータを読み込む。ステップＳ３０３において、上記読み込まれたメタデータが正当なＸＭＬで記述されていると判断された場合、ステップＳ３０４へ進み、上記ステップＳ３０１読み込まれたディレクトリデータの後に、ステップＳ３０２で読み込まれたメタデータを接続し、ステップＳ３０６において、得られたデータの全体をディレクトリデータファイルとして出力する。

【選択図】　図２

認定・付加情報

特許出願の番号	特願２０００－１０９９２３
受付番号	５００００４５８５２６
書類名	特許願
担当官	第七担当上席　　００９６
作成日	平成１２年　４月１７日

＜認定情報・付加情報＞

【特許出願人】
- 【識別番号】　000001007
- 【住所又は居所】　東京都大田区下丸子３丁目３０番２号
- 【氏名又は名称】　キヤノン株式会社

【代理人】　申請人
- 【識別番号】　100076428
- 【住所又は居所】　東京都千代田区紀尾井町３番６号　秀和紀尾井町パークビル７Ｆ　大塚国際特許事務所
- 【氏名又は名称】　大塚　康徳

【選任した代理人】
- 【識別番号】　100101306
- 【住所又は居所】　東京都千代田区紀尾井町３番６号　秀和紀尾井町パークビル７Ｆ　大塚国際特許事務所
- 【氏名又は名称】　丸山　幸雄

【選任した代理人】
- 【識別番号】　100115071
- 【住所又は居所】　東京都千代田区紀尾井町３番６号　秀和紀尾井町パークビル７Ｆ　大塚国際特許事務所
- 【氏名又は名称】　大塚　康弘

次頁無

出　願　人　履　歴　情　報

識別番号　　〔０００００１００７〕

１．変更年月日　　１９９０年　８月３０日
［変更理由］　新規登録
住　所　東京都大田区下丸子３丁目３０番２号
氏　名　キヤノン株式会社